**Debreceni Egyetem**
**Informatika Kar**

# Aktuális hálózati problémák megoldásainak vizsgálata

Szakdolgozat

**Témavezető:**
Dr. Almási Béla
Egyetemi docens

**Készítette:**
Jakab Tamás
Programtervező Informatikus

Debrecen
2009.

# Tartalomjegyzék

# 1 Bevezetés

Az internet története valahol a 60-as évek elején kezdődött, és mint oly sok napjainkban már mindennapos dolog, az Amerikai Egyesült Államok katonaságától szivárgott a civil szférába. Az akkori cél az volt, hogy egy esetleges orosz atomtámadást is túlélni képes számítógép-hálózatot hozzanak létre. Ekkor dolgozták ki azt a csomagkapcsolt protokollt, amely a mai TCP/IP architektúra ősének tekinthető. 1969-ben létrehozták az ARPANET-et, amelyet katonai célokon túl már kutatásra, egyetemek közti kommunikációra is használtak. Az évek múlásával egyre több egyetem és kutatóintézet építette ki saját hálózatát és csatlakozott a már meglévőkhöz. Az e-mailek mellett növekvő népszerűségnek örvendtek a különböző hírcsoportok is, majd 1992-ben a WWW megjelenésével bekövetkező minőségi váltásnak köszönhetően a teljesen laikus emberek számára is kezelhetővé vált az a világméretű hálózat, amit ma is internetnek hívunk.

A TCP/IP protokollcsalád hálózati rétegében működő IP protokoll nyújtja azt a címzési mechanizmust, mellyel biztosítani lehet az internetre csatlakoztatott számítógépek elérhetőségét. Az IP protokoll definícióját 1981 szeptemberében a 791-es számú RFC dokumentumban rögzítették. Ebben a dokumentumban egy IP címet 32 bit méretűnek definiáltak, ami abban az időben gyakorlatilag korlátlannak, kimeríthetetlennek látszott. Néhány éve már tudjuk, hogy ez nem így van. A NAT (Network Address Translation) és a CIDR (Classless InterDomain Routing) bevezetésével ugyan ki lehet tolni az IP címek elfogyásának dátumát, de ez csak tüneti kezelés. Az APNIC (Asia-Pacific Network Information Centre) egyik munkatársa 2011 márciusára, a CISCO szakemberei 2010 novemberére jósolják a címek elfogyását a saját szimulációjuk alapján. Az IANA becslései szerint az általuk kioszthető IP címek 2011 áprilisáig, a RIR-eknél regisztrálhatók pedig 2012 júniusáig lesznek elegendők, ha a jelenlegi ütemben halad az IP címek fogyása. Vint Cerf, akit sokan a világháló atyjának tekintenek, véleménye szerint 2012 körül merülhetnek ki véglegesen az IPv4-es címek. A helyzetet sürgetőnek nevezi, szerinte a következő generáció, az IPv6 bevezetése pusztán üzleti kérdés. Ugyan az időpontot illetően vannak eltérések, de egy dolog világosan látható: a helyzet súlyos, a probléma valós.

A világ hozzáállása azonban halogató. Mint már utaltam rá, a váltás tekinthető egyszerűen üzleti kérdésnek. A cégek nyugodt szívvel mondhatják azt, hogy egyszerűen nincs vevői

igény az IPv6-os címekre. Miért nincs igény? Mert senki nem használja. Az üzleti szféra tehát jogosan vélekedik úgy, hogy nem foglalkozik olyan szolgáltatással, amelyet senki nem használ, ráadásul a bevezetése minden tekintetben költséges. A váltás leginkább az internet szolgáltatókat sürgeti. Hogy miért? Az ok nagyon egyszerű: ha elfogynak a címek, nincs új előfizető. A helyzet felhasználói oldalról sem egyszerű. Az átlag felhasználó fél az IPv6-tól, számára a technológiai változások mindig valami bonyolult és tanulhatatlan dolgot jelentenek, ezért nem szeretik az újításokat. Ha pedig nem fél, akkor épp nem érdekli, hogy milyen protokollt használ, és az érdektelenség sem segíti elő a váltást. Továbbá nagyon sok háztartásban találhatók olyan hálózati eszközök, amelyekkel több számítógép között osztják meg az internet kapcsolatot. Ezen eszközök elhanyagolható része képes csak IPv6-os címmel működni, tehát fogyasztói oldalon is jelentkezne anyagi vonzata a váltásnak, mivel új, IPv6 kompatibilis berendezéseket kellene vásárolni. Ezen tényezők figyelembe vételével az átlag felhasználó könnyen juthat arra a döntésre, mely szerint köszöni szépen, de ő jól érzi magát a jelenlegi helyzetben. A felelősség tehát közös: a felhasználóké az, hogy hálózati eszközök vásárlása estén figyeljenek oda arra, hogy az új eszköz képes legyen az IPv6-os címek kezelésére, a gyártóké pedig, hogy olyan eszközöket kínáljanak, amelyek képesek kezelni ezt az új címzési rendszert. A szolgáltatók felelőssége pedig abban merülne ki, hogy amíg az átállás teljesen nem fejeződik be, biztosítsa a felhasználóinak az IPv4 és az IPv6 közötti átjárhatóságot. Ugyanis ha a felhasználó tudja, hogy attól, mert ő IPv6-os címet használ, még el tudja érni a világot, és a világ is el tudja érni őt, bátran fogja használni az új technológiát, hiszen nem származik belőle hátránya.

Ugyan a technológia tíz éve készen áll, a váltás folyamata nagyon nehezen indul, és még napjainkban sem túl elterjedt. Eleinte csak egyetemi kutatóintézetekben, elszigetelten, teszt jelleggel üzemeltek IPv6-os címzést használó hálózatok, és még napjainkban sincs túl nagy előrelépés. Az eddig említettek még a tyúk-tojás probléma folyományai is. A felhasználók nem látnak különbséget a két protokoll között, az ISP-knél ebből kifolyólag nincs igény, amiből pedig az következik, hogy a fejlesztők nem látnak piacot az IPv6-os alkalmazások kifejlesztésének. Ez tehát egy patt helyzet, aminek a feloldása magával hozná a váltást is. De ez csak akkor fog bekövetkezni, ha valóban elfogynak az IPv4-es címek, vagy létfontosságú szolgáltatások csak IPv6-on lesznek elérhetők, erre azonban kevesen látnak esélyt. Marad tehát minden változatlan. A kutatók kutatnak, a szakma hangoztatja a váltás szükségét, a

felhasználók pedig olvassák az új technológiáról és a bevezetésének fontosságáról szóló cikkeket az IPv4-es címmel rendelkező számítógépeiken.

Szakdolgozatom témájául azért az IPv6-ot választottam, mert úgy gondolom, hogy a váltás szükséges és elkerülhetetlen, és ha nem készülünk fel időben, akkor behozhatatlan hátrányba kerülünk a piac azon szereplőivel szemben, akik nem követték a világra egyébként jellemző halogató hozzáállást. Dolgozatomban mind elméleti, mind gyakorlati megközelítésből igyekszek körbejárni a témát.

# 2 Az IPv6

Az IPv6 egy teljesen új verziója az IP protokollnak, melyet a napjainkban is használt IPv4 utódjának terveztek. A változások elsősorban a következő területeken következtek be:

- Címtartomány kiterjesztése

  Egy IPv6-os cím mérete már 128 bit, ami lényegesen több csomópont számára teszi lehetővé az egyedi cím használatát, leegyszerűsíti az autó-konfigurálást.

- Fejrészek egyszerűsödése

  Néhány mező kikerült a szabványból, néhányat átneveztek annak érdekében, hogy csökkentsék a fejrész feldolgozásának költségét és csökkentsék az átvitelükhöz szükséges sávszélességet.

- Kiterjesztések és opcionális fejrészek támogatásának javítása

  Változásokat vezettek be a hatékonyabb továbbítás és a jövőbeni új opciók rugalmasabb bevezetése érdekében

- Adatfolyamok címkézhetősége

  Új mező bevezetésével a protokoll képessé vált a különböző kezelést igénylő adatfolyamok címkézésére és kezelésére, mint például a különböző QoS vagy Real Time szolgáltatások.

- Authentikáció és titkosítás

  Az IPv6 kiterjesztett fejlécekkel biztosítani tudja az authentikációt, az adat integritást és a bizalmas adatok kezelését.

## *2.1 Az IPv6-os fejléc*

Az IPv6 fejléc szerkezete és azok jelentése a következő:

<table>
<tr><td>Verzió</td><td>Forgalmi Osztály</td><td colspan="2">Adatfolyam Címke</td></tr>
<tr><td colspan="2">Adatmező Hossza</td><td>Következő Fejléc</td><td>Hop Limit</td></tr>
<tr><td colspan="4">Forrás Címe</td></tr>
<tr><td colspan="4">Cél Címe</td></tr>
</table>

A Verzió egy 4 bites mező, melynek értéke 6, utalva az IPv6-os fejlécre.

A Forgalmi osztály 8 bites, hasonló az előd protokoll Type Of Services mezőjéhez. Alapvetően két részből áll, a 0-5. bitek azonosítják a különböző osztályokat, a 6-7. bit pedig az úgynevezett explicit elárasztásvezérlés bitjei. Az elárasztásvezérlés lényege, hogy a router jelezni tudjon a célnak, ha elárasztást érzékel. Ekkor ugyanis a célállomás is tud jelezni a küldőnek, aki ezek után vissza tudja fogni az adást annak érdekében, hogy ne legyen csomagvesztés. Ezen két bit felépítése a következő:

| Érték | Funkció | Magyarázat |
|---|---|---|
| 00 | Nem ECT | A kapcsolat nem képes elárasztásvezérlésre |
| 01 | ECT(1) | Elárasztásvezérlés képes átvitel (1) |
| 10 | ECT(0) | Elárasztásvezérlés képes átvitel (0) |
| 11 | CE | Elárasztás észlelése |

A routerek azonos módon kezelik az ECT(1) és ECT(0) értékeket, a küldő pedig szabadon dönthet, hogy melyiket állítja be. Különbséget a szállítási protokollok tehetnek a két érték között. Az első hat bitet három részre osztották. Ezek az úgynevezett DS (Differentiated Services) bitek. A xxxxx0 formájúak szabvány által definiáltak, a xxxx11 formájúak kutatási és helyi célokra vannak fenntartva, csakúgy, mint a xxxx01, azzal a különbséggel, hogy ez utóbbit szükség szerint a jövőben bevonhatják a szabványosítás folyamatába. A szabványosítás folyamán bizonyos osztályokat a visszafelé kompatibilitás érdekében meghagytak.

Azok a csomópontok, amelyek támogatják a Forgalmi Osztály bizonyos bitjeinek speciális kezelését, megtehetik azt, hogy a tőlük származó, vagy hozzájuk érkező, vagy rajtuk keresztül haladó csomagokban megváltoztassák ezeket a biteket a speciális kezelésnek megfelelően. Amelyeknek viszont nem támogatják az adott bit sajátos kezelését, azokat figyelmen kívül, és változatlanul kell hagyniuk. Fontos még megjegyezni, hogy a felsőbb rétegbeli protokolloknak nem kell azzal foglalkozniuk, hogy egy fogadott csomag Forgalmi Osztály bitjei megegyeznek-e azokkal az értékekkel, amelyekkel a forrás elküldte őket. Egy csomóponton belül mindig a felsőbb rétegbeli protokollhoz kapcsolódó IPv6 interfésznek kell biztosítani az eszközöket az adott felsőbb protokolltól származó csomagok Forgalmi Osztály bitjeinek beállításához. Az alapértelmezett érték mind a 8 biten nulla.

A következő 20 bit az Adatfolyam Címke, amit a források arra használnak, hogy elnevezzék azokat a csomag sorozatokat, amelyeknél valamilyen speciális kezelési módot

kérnek. Egy forrás és egy cél között egyszerre több aktív folyam is lehet, és küldhetnek egymásnak olyan csomagokat is, amelyek nem részei egyetlen adatfolyamnak sem. Egy folyamot egyértelműen azonosít a forrás címe és a nem nulla adatfolyam címke. Amely folyamhoz nem tartozik címke, ott a mező értékét nullára kell állítani. A forrás által véletlenszerűen generált értéknek minden időpillanatban egyedinek kell lenni az 1-FFFFF tartományon belül. Egy meghatározott adatfolyamhoz tartozó csomagot mindig ugyanazzal a forrás, cél és adatfolyam értékekkel kell küldeni. A routerek vagy a cél állomás megvizsgálhatja, hogy ez a feltétel teljesül-e, de nem kötelező ezt megtenniük. Ha azt tapasztalják, hogy a fenti feltétel nem teljesül, akkor azt egy ICMP üzenetben közlik a feladóval. A címke élettartamában a kommunikáló felek a kapcsolat kiépítése során megegyeznek, és a címke maximális élettartamán belül sem használható fel újabb folyam azonosításához.

Az Adatmező Hossza egy 16 bites előjel nélküli egész, ami az IPv6-os adatrész hosszát adja meg bájtokban. Minden kiterjesztett és opcionális fejléc az adatfolyam része, ezért ezek mérete is beleszámít az Adatmező Hossza mező értékébe. Előfordulhat azonban, hogy az adat nem fér el 65535 bájtban. Erre az esetre dogozták ki a Jumbogramokat. Amennyiben a küldő Jumbogramot küld, az Adatmező Hossz mezőt minden esetben nullára kell állítani. Ilyenkor a hossz információt ugyanis a Hop-by-Hop fejléc fogja hordozni. Ekkor a küldőnek 32 bit áll rendelkezésére, ami azt jelenti, hogy 4,294,967,295 bájtnyi adatot tud elküldeni egy csomagban.

A Következő Fejléc mező arról szolgáltat információt, hogy milyen fejléc követi közvetlenül az IPv6 fejlécet. A következő kiterjesztett fejléceket definiálták:

- Hop-by-Hop
- Routing
- Fragment
- Destination Option
- Authentication
- Encapsulating Security Payload

A különböző kiterjesztett fejléceket a következő fejezetben részletezem.

A Hop Limit mező egy 8 bites előjel nélküli egész érték, amelyet a küldő állít be, és minden csomópont, amelyen áthalad a csomag pontosan 1-el csökkenti. A nulla Hop Limit értékkel rendelkező csomagot el kell dobni. Hasonló az IPv4 TTL mezőjéhez.

## *2.2 Kiterjesztett fejlécek*

Az IPv6-ban bizonyos hálózati rétegbeli funkciókhoz különálló fejléceket hoztak létre, amiket az IPv6 fejléc és a felsőbb rétegbeli fejléc közé lehet elhelyezni. Ezekből egy csomagban nulla, egy vagy több darab is jelen lehet. Egy kivétellel ezeket a kiterjesztett fejléceket csak a célállomás dolgozza fel. Ez a kivétel pedig a Hop-by-Hop fejléc, amit viszont minden olyan csomópont feldolgoz, amelyet a csomag az útja során érint. Ezért ennek a fejlécnek közvetlenül az IPv6 fejléc után kell következnie, és jelenlétét a Következő Fejléc mező nullás értéke mutatja. Ha egy csomópont nem ismeri fel a Következő Fejléc értékét, akkor eldobja a csomagot és egy ICMP Parameter Problem üzenetet küld a feladónak, melyben az 1-es hibakód szerepel („Unrecgonized Next Header Type Encountered”). Ahogy az előző fejezetben említettem, 6 kiterjesztett fejlécet definiáltak az IPv6-hoz. Ezeket a következő sorrendben ajánlott elhelyezni az IPv6 fejléc után:

1. Hop-by-Hop
2. Destination Option Header (a célállomáson kívül a Routing Header által meghatározott állomásoknak is szól)
3. Routing Header
4. Fragment Header
5. Authentication Header
6. Encapsulating Security Option Header
7. Destination Option Header (kizárólag a célállomásnak szól)

Ezek után következhet a felsőbb rétegbeli protokoll fejléce. A fenti felsorolástól függetlenül azonban egy IPv6-os csomópontnak el kell fogadni és meg kell kísérelni feldolgozni a fejléceket, bármilyen sorrendbe is forduljanak elő. Ez alól csak a Hop-by-Hop fejléc a kivétel, ugyanis ahogy már említettem, annak közvetlenül az IPv6 fejléc után kell állnia.

### 2.2.1 Opciók

A Kiterjesztett Fejlécek általában típus-hossz-érték(THÉ) formában kódolt Opciókat hordoznak a következő módon:

| Opció típusa | Opció hossza | Adat |
|---|---|---|

Opció típusa: 8 bites azonosító, amely azonosítja a típust (Jumbogram esetén az érték C2)

Opció hossza: 8 bites előjel nélküli egész, az adat hossza bájtokban (Jumbogram esetén az érték 4)

Adat: Változó hosszú mező, az opció típusának megfelelő adatokkal feltöltve

Amennyiben egy fejléc több THÉ hármast tartalmaz, a fogadó köteles szigorúan a felírás sorrendjében feldolgozni azokat. Az Opció típusa ezen felül még egy speciális, belső szerkezettel is rendelkezik. A felső két bitje azt írja le, hogy mit kell a csomópontnak tennie, ha nem ismeri fel az adott típust, a harmadik pedig arról szolgáltat információt, hogy az Adat mező értéke változhat-e a célhoz vezető út során. A két legmagasabb helyi értékű bit jelentése tehát a következő:

- 00 átugrik ezen a THÉ hármason, és folytatja a fejléc feldolgozását
- 01 eldobja a csomagot
- 10 eldobja a csomagot, és függetlenül attól, hogy a cél multicast cím volt-e, ICMP Parameter Problem csomagot küld 2-es hibakóddal („Unrecognized IPv6 Option Encountered”)
- 11 Eldobja a csomagot, és ha a cél nem multicast cím, akkor ICMP Parameter Problem csomagot küld 2-es hibakóddal („Unrecognized IPv6 Option Encountered”)

Ha a harmadik legmagasabb helyi értékű bitje 1-es, akkor az Adat mező tartalma út közben megváltozhat, egyébként nem. Amennyiben a csomagban Authentikációs fejléc is jelen van, az Adat rész értékét nullának kell tekinteni a csomag hitelességének ellenőrzésekor. Egy típust természetesen nem csak az alsó 5 bitje azonosít, hanem a fent említett 3 bit és a maradék 5 bit, és ennek megfelelően együtt kell őket kezelni. Különböző opciók különböző méretűek lehetnek, ezért bizonyos esetekben szükség lehet arra, hogy az adatainkat bájthatárra illesszük. Erre két fajta igazítási lehetőséget vezettek be:

- Pad1 opció: Már szerkezetét tekintve is egy speciális opció, ugyanis nincs se hossz, se adat mezője. Egy bájt beszúrására alkalmas, jelenlétét a típus mező nullás értéke jelzi.

- PadN opció: Több bájt beszúrására alkalmas. Szerkezete a következő:


Ahogy az ábrán is látszik, jelenlétét a típus mező egyes értéke mutatja. Ha N darab bájt beszúrására van szükség, a Hossz mező értékének N-2 lesz beállítva, az Adat mezőben pedig N-2 darab nullás értékű bájt fog helyet kapni.

Nézzük most egyenként a különbözőkiterjesztett fejléceket.

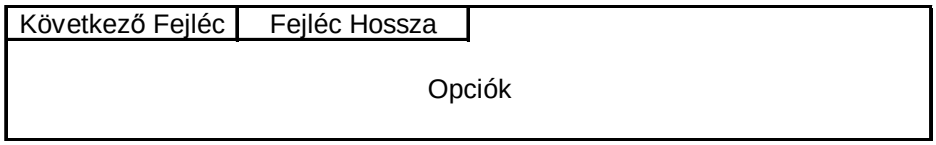
### 2.2.2 Hop-by-Hop fejléc

Olyan információkat hordoz, amelyek a célba vezető út során minden csomópont számára értékes lehet, ezért ezt mind fel is dolgozza. Az IPv6 fejléc Következő Fejléc mezőjének nullás értéke azonosítja. Szerkezete a következő:


A Következő Fejléc mező egy 8 bites érték, mely azonosítja a Hop-by-Hop fejlécet követő headert. A Fejléc Hossza szintén 8 bites, viszont nem számít bele a Következő Fejléc 8 bitje, tehát az értékét csak a fejléc hátralévő részének (Opció mező méretének) valamint a saját 8 bitjének az összege adja. Mivel ez a két mező ugyanezekkel a paraméterekkel rendelkezik a többi fejlécben is, a későbbiekben csak azokban az esetekben tárgyalom őket, ahol eltérés van az itt leírtakhoz képest. Az Opció mező mérete és szerkezete megegyezik a fentebb már bemutatottal, ezért arra nem térek ki. Tipikusan Hop-by-Hop opció a már ismertetett Pad1 és PadN igazításra szolgáló opciók.

### 2.2.3 Destination Options Header

Ez a fejléc arra szolgál, hogy kiegészítő információkat juttassunk el a cél állomásra vagy állomásokra. Ilyen kiegészítő információ lehet a már bemutatott Pad1 és PadN opció. A fejléc jelenlétére a 60-as Következő Fejléc érték utal. Szerkezete a következő:



Az Opciók mező itt is változó hosszúságú, méretére vonatkozóan itt is csak annyi van megkötve, hogy 8-nak a többszörösének kell lenni. Minden további megegyezik az előzőekben definiált Opció Mezőnél leírtakkal.
Itt jegyezném meg, hogy két lehetséges módja is van annak, hogy rendeltetési információt kódoljunk az IPv6-os csomagokba. Az egyik a Destination Header Opció mezőjét használja, a másik pedig egy különálló kiterjesztett fejlécet (például az Authentication Headert). Attól függően válasszunk e két lehetőség közül, hogy milyen viselkedésmódot szeretnénk elérni abban az esetben, amikor a cél állomás nem ismeri fel az opcionális információt. Ha például azt akarjuk, hogy eldobja a csomagot és csak akkor küldjön ICMP üzenetet, ha a célcím nem multicast, akkor bármelyik módszert választhatjuk. A döntéskor azt érdemes megfontolni, hogy melyik megoldás jár kevesebb igazítással, használ kevesebb bájtot, vagy éppen melyiket lehet gyorsabban feldolgozni. Amennyiben más viselkedésmódot szeretnénk elérni, akkor továbbra is ott van az Opció mező felső két bitje, ami a már bemutatott módon tudja befolyásolni a választ.

Ezzel a különböző fejrészek tárgyalásának a végére értem. Itt jegyezném meg, hogy van egy „különc” Következő Fejléc érték, mégpedig az 59-es. Ha ez szerepel egy Következő Fejléc mezőben, akkor az azt jelenti, hogy nincs következő fejléc, azaz semmi nem követi már azt a fejlécet. Amennyiben az Adatmező Hossz mező értéke mégis azt sejteti, hogy van még adat a csomagban, akkor azokat figyelmen kívül kell hagyni, és amennyiben a csomagot tovább kell küldeni, változatlanul kell őket hagyni.
Eddig arról beszéltem, hogyan épül fel egy IPv6-os csomag, viszont magáról az IPv6-os címről csak annyit tudunk, hogy 128 bites. A következő fejezetet ennek a 128 bitnek szentelem.

### 2.2.4 Routing fejléc

Ez a fejléc arra szolgál, hogy a csomag küldője meg tudjon határozni egy utat, ami mentén a csomagját irányítani fogják. Ebben az esetben tehát nem a forgalomirányítók döntik el, hogy merre haladjon a csomag, hanem a forrás. Ennek természetesen van hátránya is, amint majd látni fogjuk. 43-as Következő Fejléc érték azonosítja, szerkezete pedig a következő:

| Következő Fejléc | Fejléc Hossza | Routing Típusa | Hátralévő Szegmens |
|---|---|---|---|
| Típus specifikus információk | | | |

A Routing Típusa mező egy 8 bites azonosító, ami megmondja, milyen fajta forgalomirányítást kér a forrás a csomagjának. A Hátralévő Szegmens 8 bites egész, ami azt mondja meg, hogy hány csomópontot kell még a csomagnak érinteni, amíg célba nem ér. A Típus Specifikus Információk mező változó hosszúságú adathalmazt tartalmaz, melyek többnyire IPv6-os címek. Ez a mező tartalmazza például azon csomópontok IP címét, amelyeken a csomagnak át kell haladni. Ha egy csomópont olyan csomagot dolgoz éppen fel, amelyiknek nem is meri fel a Routing Típusát, akkor a csomag sorsa a Hátralévő Szegmens mező értékétől függ:

- Amennyiben a Hátralévő Szegmens értéke nulla, akkor a csomópontnak figyelmen kívül kell hagyni a Routing fejlécet, és azzal a fejléccel kell foglalkozni, amit a Routing fejléc Következő fejléc mezője meghatároz.
- Ha viszont a Hátralévő Szegmens értéke nem nulla, akkor a csomag eldobása mellett egy ICMP Parameter Problem üzenetben tájékoztatja a forrást, hogy nullás kódú kivétel történt („Erroneous Header Field Encountered”)

Amennyiben a feldolgozás után egy olyan linkre kellene küldeni, amelynek a MTU-ja (Maximum Transmission Unit – maximálisan átvihető egység) kisebb, mint a csomag mérete, akkor egy ICMP Packet Too Big üzenetben értesíti a forrást a hibáról (Az üzenet tartalmazza a hibát kiváltó link MTU-ját is.)

### 2.2.5 Routing Header Type 0

A világ kezdetben úgy gondolta, hogy jó lesz nekünk, ha megmondhatjuk, hogy merre menjenek a csomagjaink. Azóta a világnak arra is rá kellett jönnie, hogy ezzel a fejléccel viszonylag kis sávszélességgel rendelkező csomópont is képes hatalmas adatforgalmat generálni, amit akár DoS (Denial of Services) típusú támadásokhoz is felhasználhat. A 0-s típusú Routing fejléc ugyanis a következő módon épül fel:

| Következő Fejléc | Fejléc Hossza | Routing Típusa=1 | Hátralévő Szegmens |
|---|---|---|---|
| Fenntartott | | | |
| Cím 1 | | | |
| Cím 2 | | | |
| Cím 3 | | | |
| … | | | |
| Cím N | | | |

(A fenntartott 32 bitet csupa nullával kell inicializálni, és a feldolgozás során figyelmen kívül kell hagyni)

Mint látható, a fejléc egyszerre több címet is tartalmazhat, amelyeken a csomagnak végig kell haladni. A probléma ott van, hogy ugyanaz a cím akár többször is előfordulhat, és könnyen lehet olyan 0-ás típusú Routing fejlécet készíteni, aminek a segítségével elérhetjük, hogy 2 csomópont között oszcilláljon a csomagunk. Ha ügyesek vagyunk, ennek az a vége, hogy az adott két csomópont közötti forgalom megbénul, és a támadó elérte a célját. 2007-ben a CanSecWest-en Philippe Biondi és Arnaud Ebalard bemutatott egy 88-szoros erősítéses támadást ezzel a technikával. 2007 decemberében az 5095-ös számú RFC hivatalosan is érvénytelenítette ezt a típusú Routing Headert.

### 2.2.6 Fragment header

Ezt a fejlécet akkor használjuk, amikor olyan csomagot küldünk, ami nagyobb a link MTU-jánál. Ilyenkor mindig a forrás csomópontnak kell gondoskodni arról, hogy a csomagot a megfelelő méretű darabokban küldje. Az IPv6 ugyanis nem teszi lehetővé a routereknek, hogy darabolják a csomagot, mint az IPv4 esetén. Jelenlétét a 44-es Következő Fejléc érték jelzi. Felépítése:

| Következő Fejléc | Fenntartott | Fragment Offset | Fenn-tartott | M |
|---|---|---|---|---|
| Azonosító | | | | |

Minden olyan csomag, amely méretben meghaladja az adott link MTU-ját, legalább két darabra lesz bontva. Az első darab (az eredeti csomag „nem darabolható” része) tartalmazza az IPv6 Fejlécet és minden olyan kiterjesztett fejlécet, amelyet a célhoz vezető úton a csomópontoknak fel kell dolgozni (Hop-by-Hop, Routing fejléc). A második darabtól kezdve (az eredeti csomag „darabolható” része) már csak olyan információk vannak a csomagokban, amelyek csak a cél állomás számára hordoznak információt (további kiterjesztett fejlécek, adat, és egyéb, felsőbb rétegbeli protokollok fejlécei). A 8 bites Következő Fejléc ebben az esetben a nem darabolható rész első fejlécének a megfelelőjét tartalmazza. Mindkét Fenntartott mezőt (8+2 bit) nullákkal kell inicializálni, és a feldolgozás során figyelmen kívül kell hagyni. A Fragment Offset mező egy 13 bites előjel nélküli egész számot tartalmaz. Ez a szám mondja meg, hogy a fejlécet követő adat bájtokban kifejezve mennyivel van eltolva az eredeti csomag darabolható részének kezdetétől. Az M egy jelzőbit, 1-es érték esetén van még töredék, 0-ás érték esetén pedig ez volt az utolsó darabja az eredeti csomagnak. Minden egyes csomaghoz, amit darabolni kell, a forrás állomás generál egy 32 bites Azonosítót, aminek a csomag élete során egyedinek kell lenni. A csomag életébe beletartozik a szállítási idő és a töredékek összerakására szánt idő is.

Egy nagy méretű csomag darabolás után a következő szerkezettel rendelkezik:

| | | |
|---|---|---|
| Nem darabolható rész | Fragment Fejléc | Első darab |
| Nem darabolható rész | Fragment Fejléc | Második darab |
| Nem darabolható rész | Fragment Fejléc | Harmadik darab |
| | ... | |
| Nem darabolható rész | Fragment Fejléc | Utolsó darab |

Ilyenkor természetesen változik az eredeti IPv6-os fejlécünk is. Megváltozik például az Adatmező Hossza mező. Az értéke ugyanis csak az IPv6-os fejléc, a Fragment Fejléc és az első töredék méretének összegéből fog állni. A nem darabolható rész utolsó fejlécének Következő Fejléc mezője pedig a 44-es értéket fogja megkapni, jelezve ezzel, hogy darabolt csomaggal van dolgunk. Miután sikeresen feldaraboltunk egy csomagot, most nézzük meg, hogyan lesz belőle ismét egész.

Az eredeti csomag összerakása során csak azokat a darabokat használjuk, melyeknek azonos a forrás és cél címük, valamint az Azonosítójuk. Mivel a darabolás során az eredeti csomagunk

bizonyos fejléceinek mezőit megváltoztattuk, el kell végeznünk néhány számítást, hogy ismét az eredeti érték legyen mindenhol. Először a Következő Fejléc értékét állítjuk vissza. Ezt úgy tudjuk megtenni, hogy az első darab Fragment Fejlécének Következő Fejléc értékét átmásoljuk a nem darabolható rész utolsó fejlécének Következő Fejlécébe. Ezután már csak az IPv6 fejléc Adatmező Hossz értékének visszaállítása maradt. Ezt a következő képlet segítségével tudjuk meghatározni:

AH.ered = AH.els - FH.els - 8 + ( 8 * FO.utls ) + FH.utls , ahol:

AH.ered : az összerakott csomag Adatmező Hossza

AH.els : az első darab Adatmező Hossza

FH.els : az első Fragment Fejlécet követő darab hossza

FO.utls : az utolsó darab Fragment Fejlécének Fragment Offset értéke

FH.utls : az utolsó Fragment Fejlécet követő darab hossz

Miután sikeresen kiszámoltuk az értékeket, és a darabokat a Fragment Offset segítségével szépen egymás mögé illesztettük, már nem lesz jelen egy Fragment Fejléc sem a csomagban. Összeszereléskor azonban előfordulhatnak olyan helyzetek, amit a cél állomás nem tud megoldani. Ilyenek hibák például a következők:

- Még nem érkezett meg elegendő darab az eredeti csomag előállításához, de az első töredék érkezése óta már eltelt 60 másodperc. Ilyenkor két dolgot tehetünk. Ha megérkezett az első darab (sorrendben az első, és nem időrendben!), akkor az összes darab eldobása mellett egy ICMP Time Exceeded üzenetet küldünk a feladónak 1-es hibakóddal („Fragment Reassembly Time Exceeded”). Amennyiben ez nem következett be, akkor csak egyszerűen figyelmen kívül hagyjuk a megérkezett darabokat, és semmiféle hibaüzenetet nem küldünk senkinek.
- Ha a töredék hosszának számolása során nem olyan szám jön ki, ami a 8-nak egész számú többszöröse, és az M bit azt jelzi, hogy van még töredék (értéke 1), akkor ezt egy ICMP Parameter Problem üzenetben 0-ás hibakóddal („Erroneous Header Field Encountered”) jelezni kell a feladónak, és ezt a töredéket el kell dobni.
- Ha a darab hossza vagy Offset értéke nagyobb, mint 65535, illetve a vele egésszé váló csomag Adatmező Hossza szintén meghaladná ezt az értéket, akkor ezt is egy ICMP Parameter Problem üzenetben jelezzük a feladónak, mégpedig a 0-s hibakóddal.

### 2.2.7 Authentication Header

Az Authentication Header kapcsolatmentes integritás ellenőrzésre és forráshitelesítésre alkalmas. Mivel az IPv6-os csomag bizonyos fejrészeinek bizonyos mezői megváltozhatnak út közben, ezért ezzel a fejléccel nem lehet teljes védelmet biztosítani a kommunikáció során. Az Authentication Headert ezért akár az Encapsulating Security Payload Headerrel, akár különböző beágyazott módon is használhatjuk. A Következő Fejléc 51-es értéke mutatja a jelenlétét. A következő módon épül fel:

| Következő Fejléc | Adatmező Hossza | Fenntartott |
|---|---|---|
| Security Parameters Index (SPI) | | |
| Sequence Number Field | | |
| Integrity Check Value (ICV) | | |

A Következő Fejléc egy 8 bites érték, amely azonosítja a következő adatfolyamot. Értékei az IANA által meghatározott IP protokoll számok (például: 0 IPv6 Hop-by-Hop; 41 IPv6; 58 ICMPv6; 88 EIGRP; stb.). Az Adatmező Hossza egy 8 bites mező, amely a fejléc hosszát írja le szavakban (Számítástechnikai értelemben vett szavakban, azaz 32 bites egységekben.), melyből levonunk kettőt. Ha tehát egy 128 bites IVC értéket viszünk át, akkor az értéke 5 lesz. Ugyanis a fejléc első 3 szava fix, utána jön az ICV érték, ami 4 szóból áll végül levonjuk a kettőt (3+4-2=5). A 16 bites fenntartott mezőt itt is nullákkal kell inicializálni, feldolgozáskor pedig figyelmen kívül kell hagyni. Az SPI mező egy tetszőleges 32 bites érték, melyet a cél állomás arra használ, hogy azonosítsa a csomaghoz tartozó biztonsági kapcsolatot (SA). Ha a két fél kommunikációja során unicast címeket használ, akkor az SPI önmagában is alkalmas arra, hogy azonosítsa a biztonságos összeköttetést. Multicast típusú kommunikáció esetén a helyzet egy kissé komplikáltabb. Ilyenkor általában nem a különböző csomópontok határozzák meg ezt az értéket, hanem egy külön Kulcs Szerver. Ez a szerver önhatalmúlag dönt az egyes csoportok által használandó SPI értékekről, amit azok nem utasíthatnak vissza. A csoport tagjai ettől függetlenül természetesen még dönthetnek az unicast kommunikációjuk során használt kulcs értékekről, ahhoz ugyanis semmi köze a Kulcs Szervernek. A kapcsolatokra vonatkozó információkat a szerverek az úgynevezett SAD táblában tárolják. A SAD tábla bejegyzései többek között az alábbiakból épülnek fel:

- SPI

- Sequence Number Counter: Az AH és az ESP fejlécek Sorszám mezőjének generálásához. Mérete 64 bit
- Sequence Number Owerflow: túlcsordulás jelzésére
- Anti-Replay Window: ismételt keretek detektálásához
- Különböző, AH specifikus adatok
- Különböző, ESP specifikus adatok
- SA élettartama
- IPSec protokoll üzemmódja
- Út MTU-ja

Amikor egy hoszt a SAD táblában keres, több bejegyzést is kaphat eredményül. Azt, hogy melyik lesz a valóban keresett, a következő algoritmus segítségével határozzák meg:

- Ha az SPI,forrás cím és cél cím is megegyezik, akkor a bejövő AH csomagot ennek a SAD bejegyzésnek a segítségével dolgozza fel. Ha nem, akkor lép a következő pontra.
- Ha az SPI és a forrás cím egyezik csak, akkor ezt a SAD bejegyzést használja. Ha nem, akkor a harmadik lépés következik
- Ekkor már csak a SPI értékek egyezését követeljük meg. Ha sikerült az illesztés, akkor ezzel a SAD bejegyzéssel dolgozza fel a csomagot, ellenkező esetben az esemény naplózása mellett eldobja a csomagot.

Az egyes implementációk bármilyen módon megvalósíthatják a keresést, azzal a megkötéssel, hogy kívülről ezt a viselkedésmódot kell, hogy mutassák. A következő mező a 32 bites Sequence Number mező, ami funkcióját tekintve egy számláló. Ebből kifolyólag a küldő félnek minden esetben növelni kell az értékét 1-el. A multicast alapú kommunikáció során az AH nem biztosít semmiféle szinkronizációt az értékére vonatkozóan, ugyanakkor az AH implementációk mindegyikének képesnek kell lenni elvégezni a következőket:

- Sorszám generálása: A sikeres SA felépítését követően a küldő nullás értékkel inicializálja a mezőt. Küldés előtt növeli a sorszám értékét eggyel, és a kapott érték alsó 32 bitjét beilleszti a Sequence Number mezőbe (Ezt az Extended Sequence Number miatt fontos megjegyezni, ami 64 bites.). Ha az Anti-Replay engedélyezve van, akkor a beillesztés előtt ellenőrizni kell, hogy az érték nem-e fordult már elő. A küldő ugyanis nem küldhet olyan csomagot, mely a Sequence Number értékét túlcsordulttá teszi. Az Anti-Replay

kikapcsolása csak multicast esetben javasolt, ott ugyanis nincs beépített mechanizmus a szinkronizációra.

- Sorszám ellenőrzése: Ha a fogadó oldalon nincs bekapcsolva az Anti-Replay, akkor bejövő csomag esetén nincs ellenőrzés. Ugyanakkor a küldő félnek azt kell feltételezni, hogy a fogadó elvégzi az ellenőrzést. Annak érdekében, hogy a küldő ne végezzen felesleges ellenőrzéseket, az SA kiépítése során ajánlott tájékoztatni őt arról, hogy a fogadó nem támogatja az Anti-Replayt. Amennyiben viszont támogatja, az SA felépítése során neki is inicializálni kell a Sequence Number értékét. Miután meghatároztuk, hogy melyik SA-hoz tartozik a csomag, tanácsos rögtön ellenőrzi a sorszámot, elősegítve ezzel a duplikációk lehető leggyorsabb detektálását. A duplikációk kiszűrését egy csúszó ablak segítségével végzik. Az ablak jobb oldala a kommunikáció során már használt legnagyobb érték. Amennyiben a fogadott csomag sorszáma az ablak bal oldalán kívül esik, a csomagot el kell dobni. Ha a sorszáma beleesik az ablakba, akkor ellenőrizni kell, hogy fogadtunk-e már ilyen sorszámú csomagot. Amennyiben a csomag nem duplikáció, vagy az ablak jobb oldalára esik, akkor a feldolgozás folytatódhat az ICV érték ellenőrzésével. Ha az ICV nem egyezik, akkor a csomagot el kell dobni, és naplóbejegyzést kell készíteni. A fogadó ablakot csak abban az esetben kell módosítani, ha az ICV érték is valósnak bizonyult. Meg kell még jegyezni, hogy a fogadó ablak méretére csak alsó korlát van megadva, mégpedig 32 bitben. A fogadó fél ugyanis saját hatáskörben dönt az ablak méretéről, hiszen a napjainkban már létező sok gigabites átviteli sebesség más méreteket igényel, mint például egy néhány megabites otthoni ADSL kapcsolat.

A nagy sebességű kapcsolatok támogatása érdekében definiáltak egy kiterjesztett sorszámozási megoldást is. Ekkor a számláló mérete 64 bit, aminek ugyan csak az alsó 32 bitje szerepel a fejlécben, viszont az ICV érték számításakor figyelembe van véve.

Az ICV mező egy változó hosszúságú mező, amely egy ellenőrző összeget tartalmaz. A mező méretének kötelezően oszthatónak kell lenni 32-vel, amit a különböző igazításokkal lehet elérni. Mivel az egyes fejlécek mezőinek értékei változhatnak, ezért az ellenőrző összeg csak bizonyos mezők értékeiből számítódik ki. Ahhoz, hogy meghatározzuk ezeket a mezőket,

először el kell különíteni azokat a mezőket, amelyek nem változhatnak, amelyek változhatnak, de a változás megjósolható, és azokat, amelyek változhatnak, és nem számítanak bele

Nem változhatnak:

- Verzió
- Adatmező Hossza
- Következő Fejléc
- Forrás cím
- Cél cím (Routing Fejléc nélkül)

Változhatnak, de ez megjósolható:

- Cél cím (Routing Fejléccel)

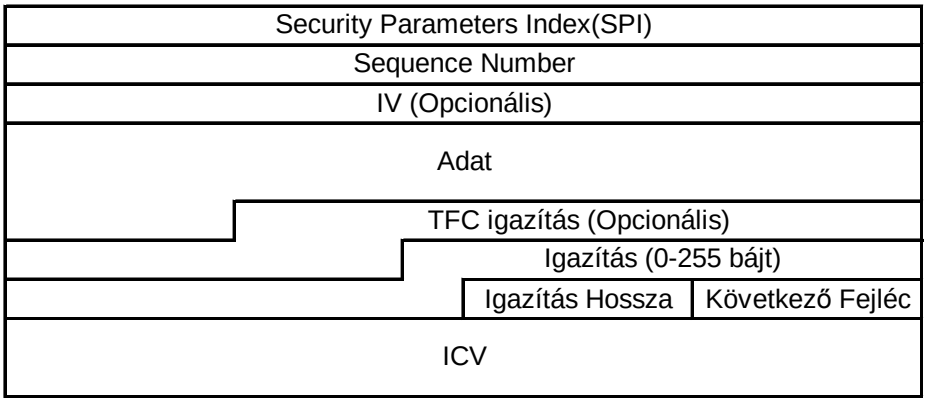
Változhatnak, és nem számítanak bele: (értéküket nullázni kell)

- Differentiated Services bitjei (6 bit)
- ECN bitek (2 bit)
- Adatfolyam Címke
- Hop Limit

A Hop-by-Hop és Destination Options Fejlécek tartalmaznak 1-1 bitet, amely jelzi, hogy az általuk hordozott adat változhat-e út közben. Amennyiben olyan adatot tartalmaznak ezek a fejlécek, amelyek változhatnak, akkor ezeknek a mezőknek az értékét csupa nullának kell tekinteni miközben az ellenőrző összeget számoljuk. A cél állomás a meghatározott adatok alapján szintén kiszámolja az ellenőrző összeg értékét, és ha ez egyezik a csomagban szereplővel, akkor a csomag hiteles. Amennyiben az ellenőrző összegek eltérnek, akkor a csomag nem hiteles, és el kell dobni. Ekkor egy naplóbejegyzés is készül az eseményről.

### 2.2.8 Encapsulating Security Payload Header

Szolgáltatásait tekintve hasonló az Authentication Headerhez. Képes titkosításra és integritás ellenőrzésre is. Szerkezete a következő:




Az SPI és Seqence Number mezőkre ugyanazok igazak, mint az AH esetében. Ebben a fejlécben az adat mező rendelkezhet egy sajátos belső szerkezettel: Bizonyos esetekben ugyanis a titkosító algoritmus elhelyez az adatok elé egy inicializációs vektort (IV). Az ábra ezt az esetet szemlélteti. Az adat mezőnek minden esetben egész számú bájtot kell tartalmaznia, az esetleges IV értéket pedig szerves részeként kell kezelni. Mivel a felsőbb rétegbeli protokoll fejlécének 8-as bájthatáron kell kezdődni, ezért szükség lehet igazításra. Ennek mérete maximum 255 bájt lehet. De előfordulhat olyan eset is, amikor a titkosító algoritmus a nyers szöveg méretére tesz valami megkötést. Ekkor a megfelelő igazítással tudunk ennek eleget tenni. A titkosító algoritmustól függetlenül előfordulhat, hogy a titkosított szöveg nem illeszkedik 4 bájtos határra, és ilyenkor is szükséges az igazítás használata. Ha több bájtot kell így beszúrni, azokat 1 bájtos, előjel nélküli egész számok sorozatával inicializálják (az első bájtot például az 1-es értékkel, a másodikat 2-vel, a harmadikat 3-mal, stb… a lényeg, hogy egy monoton növekvő számsor legyen.). Van még egy opcionális mező, amivel szintén lehet igazítani. Az adatfolyam részét képezi ugyanis egy Adatfolyam Bizalmasság (TFC) mező, ami arra hivatott, hogy elrejtse az adatfolyam jellegzetességeit. Az Igazítás utáni mező azt mondja meg, hogy hány bájtot szúrtunk be, ezért 0 és 255 között vehet fel értéket. Ez a mező kötelezően szerepel, és abban az esetben, amikor nem szúrtunk be egyetlen bájtot sem, nullás értéket kell kapnia. A Következő Fejléc egy 8 bites mező, ami az Adat mezőben lévő, következő fejlécet azonosítja. Értékei megegyeznek az Authentication Header azonos mezőjével, azaz az IANA által meghatározott protokoll azonosítókkal. Annak érdekében, hogy az eredeti adatfolyamot még jobban álcázhassuk, úgynevezett dummy (báb, vagy utánzat) csomagokat is létre hozhatunk. Ezeket a csomagokat a Következő Fejléc 59-es értékéről ismerhetjük fel (az 59-es fejléc a „nincs következő fejléc”

jelentéssel bír). Az AH-val ellentétben az ICV mező itt opcionális, de jelentése megegyezik azzal.

## *2.3 Az IPv6-os cím*

Amint már említettem, az IPv6-os címek 128 bites hálózati rétegbeli címek, melyek egy vagy több interfészt azonosíthatnak. Három fajtájuk létezik:

- Unicast : Egyetlen interfészt azonosít, egy unicast címre küldött csomagot a megadott címmel rendelkező interfész fogja megkapni.
- Anycast : Interfészek egy csoportját azonosító cím. Ezek az interfészek általában különböző csomóponthoz tartoznak. Egy anycast címre küldött csomag a forgalomirányító protokoll távolságfogalmának megfelelően a legközelebbi interfésznek lesz elküldve.
- Multicast : Interfészek egy csoportját azonosító cím. Egy multicast címre küldött csomagot a csoport minden tagja meg fog kapni.

Az IPv6 nem értelmez broadcast címet, ezt a funkciót a multicast cím látja el.

Az eddigiekből kitűnhet, hogy címzés tekintetében az interfész és a csomópont más jelentéssel bír. Ez azért van, mert az IPv6-os címeket mindig interfészekhez, nem pedig csomópontokhoz rendeljük. Egy csomópont rendelkezhet ugyanis több interfésszel is (természetesen az interfész is rendelkezhet több címmel, sőt, speciális esetekben akár egy címet több interfészhez is hozzárendelhetünk. Ehhez azonban speciális implementáció szükséges, és terhelés-elosztásnál kaphat nagy szerepet).

Az IPv6-os címeket hexadecimális számjegyek sorozataként jelenítjük meg, az alábbi három konvenció szerint:

1. általában x:x:x:x:x:x:x:x formában szokás megjeleníteni, ahol az x egy 16 bites hexadecimális szám. Például:
    a. 2001:15c0:65ff:0251:0000:0000:0000:0002
    b. 2001:15c0:65ff:251:0:0:0:2 (Megjegyzés: nem kötelező kiírni a vezető nullákat, de legalább egy számjegynek szerepelni kell minden mezőben. Kivétel a következő pont:)
2. Mivel gyakran előfordulnak hosszú nullás sorozatok, ezért bevezettek egy speciális jelölést: a „::” sztring egy vagy több nullákból álló 16-bites sorozatot helyettesít. Egy címben legfeljebb egyszer fordulhat elő, és bárhol használható. Például:

a. Unicast cím: 2001:15c0:65ff:251:0:0:0:2, melynek tömörített formája: 2001:15c0:65ff:251::2
b. Multicast cím: FF01:0:0:0:0:0:0:101, melynek tömörített formája: FF01::101
c. Loopback cím: 0:0:0:0:0:0:0:1, melynek tömörített formája: ::1
d. Nem meghatározott cím: 0:0:0:0:0:0:0:0, melynek tömörített formája: ::

3. Létezik egy „kevert” felírási mód is. Ekkor az utolsó négy bájtot pontozott decimális alakban írják fel. Például:
   a. 0:0:0:0:0:0:78:131:45:203; tömörítve: ::78.131.45.203
   b. 0:0:0:0:0:FFFF:78.131.45.203; tömörítve: ::FFFF:78.131.45.203

A prefixekre vonatkozóan nincs eltérés az IPv4-es címekhez képest, vagyis ugyanúgy ip-cím/prefix-hossz formában írandók. Az IP-cím az eddig meghatározott felírási módok bármelyikével megadható, a prefix hossz pedig egy decimális szám. A következő táblázat a különböző címek prefixeit mutatja:

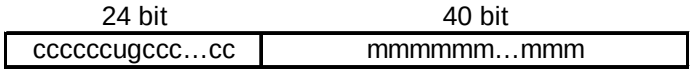
| Cím típusa | Bináris prefix | IPv6-os megjelenítése |
|---|---|---|
| nem meghatározott | 00….1 (128 bit) | ::/128 |
| Loopback | 00….0 (128 bit) | ::1/128 |
| Multicast | 11111111 | FF00::/8 |
| Link-Local unicast | 1111111010 | FE80::/10 |
| Global unicast | Minden egyéb | |

Az anycast címek az unicast címtérből vannak kiválasztva, és szintaktikailag nem lehet őket megkülönböztetni.
Ezek után nézzük egyenként a különböző típusú címeket.

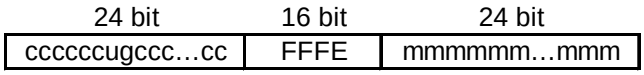
### 2.3.1 Unicast címek

Az Unicast címek a CIDR-el használt IPv4-es címekhez hasonló felépítésűek. Többféle unicast cím is létezik, mint például globális unicast, link local unicast, vagy éppen a beágyazott ipv4-es ipv6 unicast cím. Ezen felül a csomópontok is eltérő információkat tudnak kinyerni egy címből. Vannak olyanok, akik semmit nem tudnak a címek belső szerkezetéről, és vannak olyanok is (például egy router), akik legalább egy subnet prefixre és egy interfészazonosítóra fel tudják azt bontani.
Az Interfész Azonosító arra szolgál, hogy egy linken azonosítsa az interfészt, ezért ezeknek egyedinek kell lenni az adott subnet prefixen belül (az interfész azonosító egyedisége független az IPv6-os cím egyediségétől). Előfordulhat, hogy az interfészazonosítót az

adatkapcsolati rétegbeli címből határozzák meg. Ez az úgynevezett EUI64-es címzés. Amennyiben az adatkapcsolati rétegben az IEEE 802-es MAC címeket használnak, akkor a MAC cím 48 bitje és az interfészazonosító 64 bitje közötti méretkülönbséget át kell hidalni. Mivel a MAC cím két részből áll, kézenfekvőnek tűnik, hogy osszuk azt két egyforma méretű részre, és a két rész közé szúrjuk be a hiányzó két bájtot. A két bájt pedig az FFFE vagy az FFFF lesz. Amennyiben az IEEE EUI-64-es címzést használják, azt kiegészítés nélkül fel tudják használni az Interfész Azonosító előállítására. Az azonosítók ekkor a következőképpen épülnek fel:




EUI-64-es közeg-hozzáférési címmel




MAC-címmel

Az ábrán a „c” jelöli a gyártó azonosító bitjeit, az „m” az eszköz azonosító bitjeit. Mint látható, a gyártó azonosító 2 bitje speciális jelentéssel bír. Az „u” bit az úgynevezett universal/local bit, melynek 0-ás értéke jelzi, hogy a cím globálisan egyedi, az 1-es érték pedig arra utal, hogy azt csak helyi hatáskörben lehet egyedinek tekinteni. A „g” bit jelöli, hogy a cím egy unicast vagy egy multicast cím-e. Amennyiben értéke 0, úgy a cím unicast, ha pedig 1, akkor a cím multicast. Mivel azonban ezek az értékek gyárilag 00-ra vannak állítva (globális unicast címet jelölve), ezért az IEEE EUI-64-es címekből ezek megfelelő beállításával készíthetünk IPv6-os címet. A szabály pedig a következő: Az „u” bitet minden esetben az ellentettjére változtatjuk, a „g” bitre vonatkozóan pedig nincs megkötés. Ha unicast címet szeretnénk, akkor értékét nem változtatjuk, ha pedig multicast címhez akarunk hozzájutni, akkor 1-re állítjuk azt. A példa kedvéért tekintsük a következő MAC címet: 00-0B-6A-F0-6B-D8. Ekkor a következő, 64 bites interfész ID-t tudjuk belőle képezni: 020B-6AFF-FEF0-6BD8.

### 2.3.1.1 Nem meghatározott cím

A **nem meghatározott cím** csupa nullából áll, amit egyetlen csomóponthoz sem lehet hozzárendelni. Ez a cím nem szerepelhet egyetlen IP csomag fejlécében sem, és a routerek sem továbbíthatnak ilyen csomagokat.

### 2.3.1.2 Loopback cím

A 0:0:0:0:0:0:0:1 unicast címet hívják **Loopack címnek**, melyet a csomópontok arra használnak, hogy saját maguknak küldjenek vele csomagot. Csak valamilyen logikai interfésznek lehet ez a címe, és ez sem lehet olyan csomag forráscíme, ami elhagyja az adott csomópontot.

### 2.3.1.3 Globális unicast címek

Az általános megjelenésük a következő:

| n darab bit | m darab bit | 128-n-m darab bit |
|---|---|---|
| global routing prefix | subnet ID | interfész ID |

Ahol a global routing prefix azonosítja az alhálózatok/linkek egy csoportját, a subnet ID a csoporton belül azonosítja az alhálózatot, az interfész ID-re pedig az előzőekben leírtak igazak. A 000-val kezdődőeket kivéve minden unicast cím 64-bites interfészazonosítóval rendelkezik. A 000-val kezdődőekre ez a megszorítás természetesen nem érvényes.

**IPv6-os címek beágyazott IPv4-es címmel**

Két módja van annak, hogy IPv6-os cím alsó 32 bitjébe IPv4-es címet építsünk. Az egyik az „IPv6 kompatibilis IPv4”, a másik pedig az „IPv4-mapped IPv6”. A következő részben ezeket ismertetem.

**IPv6 kompatibilis IPv4**

Ezt a megoldást azért fejlesztették ki, hogy elősegítsék az átmenetet a két verzió között. A formátuma a következő:

| 80 bit | 16 bit | 32 bit |
|---|---|---|
| 0000……0000 | 0..0 | IPv4-es cím |

A címben szereplő IPv4-es címnek természetesen egyedinek kell lenni.
Mivel ezt a módszert már nem használják, ezért csak a teljesség kedvéért került ide.

**IPv4-mapped IPv6**

Ez a módszer beágyazva tartalmazza az ipv4-es címet, azt a látszatot keltve, mintha valójában az egy IPv6-os cím lenne. Felépítése a következő:

| 80 bit | 16 bit | 32 bit |
|---|---|---|
| 0000……0000 | FFFF | IPv4-es cím |

Az IPv4-es címtől ebben az esetben is megköveteljük az egyediséget.

#### 2.3.1.4 Link Local Unicast címek

A Link-Local címeket csak az adott linken belül lehet használni, formájuk a következő:

| 10 bit | 54 bit | 64 bit |
|---|---|---|
| 1111111010 | 0 | Interfész ID |

Az ilyen alakú címeket többek között autókonfigurációra, szomszédság felderítésre használják, vagy esetleg olyan környezetben, ahol nincs jelen router. Továbbá a routerek nem továbbítanak másik linkre olyan csomagokat, amelynek cél vagy forrás címe Link Local cím.
Mára a Site Local címek használata sem támogatott, azonban a teljesség kedvéért essen pár szó róla is. Felépítése a következő:

| 10 bit | 54 bit | 64 bit |
|---|---|---|
| 1111111011 | Subnet ID | Interfész ID |

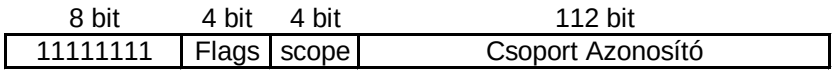
Mivel az új implementációknak a Site Local címeket nem kell támogatniuk, ezért ezt a prefixet Globális Unicast prefixként kell kezelniük.

### 2.3.2 Anycast címek

Az anycast címek olyan címek, amelyek egynél több interfészhez vannak hozzárendelve. Az anycast címre küldött csomagok a forgalomirányító protokoll által legközelebbinek kinevezett csomópontra lesznek továbbítva. Mint már korábban említettem, az anycast címek az unicast címtérből lesznek lefoglalva, és tőlük szintaktikailag sehogy sem lehet megkülönböztetni. Egy unicast címből ugyanis úgy lesz anycast cím, hogy egyszerre több fizikai interfésznek is ugyanazt a címet állítjuk be. Ilyenkor azonban ezeket az interfészeket úgy kell konfigurálni, hogy tudják magukról, hogy anycast címmel rendelkeznek. Minden kiosztott anycast cím esetén létezik egy P leghosszabb prefix, ami azonosít egy tartományt, amin belül az összes hozzá tartozó interfész megtalálható. Ezen a P-vel azonosítható tartományon belül ezt az anycast címet a forgalomirányítóknak különálló bejegyzésként kell kezelni. A P tartományon kívül ezt a címet a P-hez tartozó forgalomirányító tábla bejegyzésével kell összekapcsolni.
Legrosszabb esetben a P érték a null prefix. Ebben az esetben az anycast címet az egész interneten különálló routing tábla bejegyzéssel kell kezelni, ami egy elég erős korlátot ad az ehhez hasonló anycast címek támogathatóságára vonatkozóan.

### 2.3.3 Multicast címek

Interfészek egy csoportját azonosítják, az anycast címekkel ellentétben azonban itt minden interfész megkapja a csomagot, és egy interfész akárhány multicast címmel rendelkezhet.
Szerkezete a következő:




A bináris 11111111-el kezdődő címek tehát multicast címek.

A Flags mezőt 4 flag határozza meg:


- Az első bit fenntartott, minden esetben 0-val kell feltölteni.
- A T bit határozza meg, hogy az adott multicast cím úgynevezett „Well-Known”-e, azaz állandóan létező cím. Ezeket az IANA osztja ki.
  - Ha értéke 0, akkor Well-Known
  - Ha értéke 1, akkor ez dinamikusan létrehozott multicast cím
- A P bit azt mondja meg, hogy a multicast cím a hálózati prefixre támaszkodva lett-e meghatározva.
  - Ha értéke 0, akkor nem a hálózati prefix alapján határozták meg
  - Ha értéke 1, akkor igen. Ebben az esetben a T értéknek is 1-nek kell lenni, különben az ott leírtaknak megfelelően kell beállítani az értékét.
- Az R bit azt definiálja, hogy a multicast address magában foglalja-e az RP (Rendezvous Point) címét.
  - Ha értéke 1, akkor igen. Ekkor P-t 1-re kell állítani, ami a T értékét is meghatározza. Ekkor a prefix FF70::/12 lesz.
  - Ha értéke 0, akkor a cím nem foglalja magában az RP-t.

A Scope mező különböző értékével korlátozni tudjuk a multicast address hatáskörét. Értékei a következők lehetnek:

0. fenntartott
1. Interface-Local Scope
2. Link-Local Scope
3. fenntartott
4. Admin-Local Scope
5. Site-Local Scope
6. szabad
7. szabad
8. Organization-Local Scope
9. szabad
A. szabad
B. szabad

C. szabad

D. Globális Scope

E. Fenntartott

Az Interface-Local Scope egyetlen interfészt foglal magában, és csak Loopback alapú multicast kommunikációkor hasznos.

A Link-Local Scope a megfelelő unicast címhez hasonló topológiájú régiót foglalja magában.

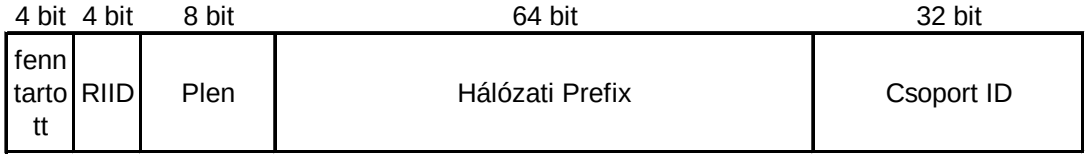
Az Admin-Local Scope a legkisebb hatáskör, amit manuálisan kell konfigurálni.

A Site-Local címeket arra tervezték, hogy egyetlenegy „site”-ot foglaljanak magukba.

Az Organization-Local hatáskört azért hozták létre, hogy a különböző szervezetekhez tartozó helyeket összefogja.

A szabad jelzéssel rendelkezőket azért hozták létre, hogy az adminisztrátorok saját hatáskörükben létrehozhassanak régiókat.

A Group ID az adott scope-on belüli multicast csoportokat azonosítja. Belső szerkezete a következő:




A fenntartott mezőnek kötelezően nullának kell lenni.

P=1 esetén a Plen mező jelzi, hogy a Hálózati Prefix mező értékéből hány bit azonosítja az alhálózatot.

A Hálózati Prefix mező azonosítja a multicast címet tartalmazó unicast alhálózatot. Ha P=1, akkor ez a mező tartalmazza a multicast alhálózatot. Minden olyan bitet, amelyik nem szükséges ennek a meghatározásához, fel lehet tölteni nullákkal is.

A Csoport ID a multicast csoportot azonosítja, függetlenül attól, hogy az permanens vagy tranziens-e az adott scope-on belül. A 101-es Group ID például az NTP szervereket azonosítja. A 2375-ös RFC-ben található egy felsorolás a multicast címekről, néhány példa:

Adott scope-on belül érvényesek:

FF02:0:0:0:0:0:0:1 a link összes csomópontja

FF02:0:0:0:0:0:0:2 a link összes routere

FF05:0:0:0:0:0:0:2 a site összes routere

FF02:0:0:0:0:0:0:D a link összes PIM routere

Bármilyen scope-on belül érvényesek:

FF0X:0:0:0:0:0:0:101 Network Time Protocol(NTP)

FF0X:0:0:0:0:0:0:109 MTP Multicast Transport Protocol

Az eddig elmondottakon felül igazak még a következők is. A multicast címek nem lehetnek forrás címek, és nem jelenhetnek meg Routing fejlécben sem. A routereknek nem kell továbbítani a célként megjelölt multicast cím Scope mezője által mutatott scope-on kívül eső multicast csomagokat. A csomópontok nem készíthetnek olyan multicast csomagot, amelynek a Scope mezője nullás értéket tartalmaz, ha ilyen csomagot fogadnak, akkor azt némán el kell dobniuk. Továbbá olyan csomagot sem hozhatnak létre, amelynek a Scope mezője a fenntartott F értéket tartalmazza, viszont fogadáskor ezt úgy kell kezelni, mintha a mező értéke a Globális Scope-re jellemző E értéket tartalmazná.

IPv6-os környezetben az interfészek többféleképpen is hozzájuthatnak érvényes címhez. Az egyik ilyen megoldás az állapotmentes autókonfiguráció. Ez a művelet akkor veszi kezdetét, amikor aktiváljuk a linket, és csak azokon a linkeken megy végbe, amelyek támogatják a multicast kommunikációt. A csomópontok első lépésben generálnak egy link-local címet a már bemutatott link-local prefix és az interfész azonosítójuk segítségével. Mielőtt azonban ezt a címet hozzárendelnék valamelyik interfészükhöz, ellenőrizni kell, hogy ez a cím valóban egyedi-e az adott linken. Ezt úgy teszik meg, hogy egy úgynevezett Szomszédság Kérést (Neighbor Solicitation) küldenek szét a linken, ami tartalmazza a próbaképpen legenerált címet. Amennyiben erre a kérésre egy Szomszédság Hirdetés (Neighbor Advertisement) érkezik, akkor ezt a címet már használják. A Szomszédság Kérés és Szomszédság Hirdetés üzeneteket egy későbbi fejezetben részletesebben fogom tárgyalni.

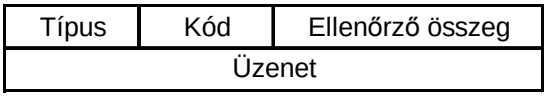
Amennyiben egy próba címről bebizonyosodik, hogy használják, akkor az autókonfiguráció folyamata megáll, és manuálisan kell beállítani az interfészt. Ha viszont egyedinek bizonyult, akkor ezt a címet hozzárendeli az interfészhez. Ekkor már tud IP alapon kommunikálni a szomszédaival. Ahhoz azonban, hogy az internetet is elérhesse, először találnia kell egy routert, akitől megtudhatja a hálózati prefixét. Amennyiben talál egy routert, az egy Router Advertisement üzenetben elküldi a cím beállításához szükséges további információkat a hoszt számára. A routerek ezeket az üzeneteket periodikusan szétküldik a hálózaton, amennyiben viszont a hoszt ezt nem tudja kivárni, egy Router Solicitation üzenet küldésével azonnal kiválthatja azt. Mivel a routerek időről időre elküldik ezeket az információkat, a hosztok mindig naprakészek lesznek a körülöttük lévő világról, és változás esetén újra tudják magukat

konfigurálni (természetesen ügyelve arra, hogy a címek minden esetben egyediek maradjanak). Az újrakonfigurálás igénye felmerülhet például azért is, mert lejárt az adott cím élettartama.

Ha egy hoszt nem talál routert a szomszédai között, akkor a DHCPv6 segítségével még mindig meg tudja határozni az interfésze számára szükséges információkat. Az első lépés ebben az esetben is az, hogy meghatározza a hoszt a link-local címét. Ezután a következő lépés, hogy a DHCP szerver címét meghatározza, majd UDP-n keresztül folytatott kommunikációval beszerezze a szükséges információkat. A szerver címének meghatározása érdekében definiálták a FF02::1:2 és a FF05::1:3 multicast címeket, a port szám, amin a DHCP szerverek hallgatóznak pedig az 547-es (a klienseké az 546-os port). Az üzenetváltások az IPv4-hez hasonló elvet követnek.

# 3 ICMPv6

Az ICMPv6-os protokollt az IPv6-os csomópontok használják különböző hibajelzésekre és diagnosztikai célokra. Az ICMPv6 szerves részét képezi az IPv6-nak, amit minden csomópontnak implementálni kell. Minden ICMPv6-os csomagot egy IPv6 fejléc és egy vagy több kiterjesztett fejléc előz meg. Az ICMPv6-os üzeneteket az 58-as Következő Fejléc érték jelzi. Az ICMPv6-os üzenetek szerkezete a következő:




A típus mező határozza meg, hogy milyen típusú információt szállít az üzenet. A többi mező formája ettől az értéktől függ.

A kód mező segítségével finomítani lehet az üzenet szerkezetét és jelentését is.

Az Ellenőrző Összeg arra szolgál, hogy bizonyos hibákat észre tudjunk venni.

Az ICMPv6-os üzeneteket két nagy csoportra tudjuk bontani. Az egyik a hibaüzenetek, a másik pedig a tájékoztató jellegű üzenetek. A kettő közötti különbséget a típus mező legmagasabb helyi értékű bitjének az értéke adja. A hibaüzenetek esetében az értéke 0, az informális üzenetek esetében pedig 1. Ebből adódik az is, hogy a hibaüzenetek 0 és 127-es típusszámot kaptak, a tájékoztató jellegűek pedig 128-255-igterjedő sorszámot.

Jelenleg a következő típusok vannak szabványosítva:

Hibaüzenetek:

1. Destination Unreachable (Cél nem elérhető)
2. Packet Too Big (Csomag méretet túl nagy)
3. Time Exceeded (Időtúllépés)
4. Parameter Problem (Paraméter probléma)

100. Magán jellegű kísérletezésre fenntartva

101. Magán jellegű kísérletezésre fenntartva

127. Fenntartva későbbi bővítés céljából

Tájékoztató jellegű üzenetek:

128. Echo Request (Visszhang kérés)

129. Echo Reply (Visszhang válasz)

200. Magán jellegű kísérletezésre fenntartva

201. Magán jellegű kísérletezésre fenntartva

255. Fenntartva későbbi bővítés céljából

Annak a csomópontnak, amelyik ICMPv6 üzenetet küld, meg kell határoznia mind a forrás, mind a cél címét az IPv6-os fejlécből, mielőtt kiszámíthatná az ellenőrző összeget. Ha a csomópontnak egynél több unicast címe van, akkor a következőképpen kell eljárnia:

- Ha az üzenet válasz valamelyik unicast címére küldött üzenetre, akkor a válasz üzenet forrás címének ugyanannak a címnek kell lenni.
- Ha az üzenet valamilyen más címre küldött válasz (multicast, anycast, vagy olyan unicast, ami nem az adott csomóponthoz tartozik), akkor a válasz üzenet forrás címének egy olyan unicast címnek kell lenni, amelyik az adott csomóponthoz tartozik.

Az ellenőrző összeg egy 16 bites érték, ami nem más, mint az egész ICMPv6 üzenet egyes komplemensének az összegének az egyes komplemense. Mielőtt ezeket a műveleteket végrehajtanánk, az egész mezőt fel kell tölteni nullákkal. Miután összeállítottuk az üzenetet, és elküldtük, a címzett a fogadáskor a következő módon jár el. Ha egy hibaüzenetnek nem ismeri fel a típusát, akkor továbbadja a csomagot egy felsőbb rétegbeli protokollnak. Ha ugyanez egy tájékoztató jellegű csomaggal történik, akkor azt némán eldobja. Ha a felsőbb rétegbeli protokoll számára csak úgy tudja átadni a csomagot, hogy egy internet rétegbeli protokoll közreműködésére is szükség van, akkor a szükséges információk az eredeti csomagból kinyerhetők, ami az ICMPv6 üzenet törzsében megtalálható. Ezután ki lehet választani a megfelelő protokollt, ami majd kezeli a felmerült hibát. Abban az esetben, amikor a felsőbb rétegbeli protokoll nem határozható meg, az ICMPv6 üzenetet némán el kell dobni, miután az IPv6-os protokoll feldolgozta azt.

ICMPv6 hibaüzenet nem keletkezhet a következő események hatására:

- ICMPv6 hibaüzenet hatására
- ICMPv6 átirányítás hatására (a routerek átirányíthatják a forgalmat, ha jobb útvonalat tudnak.)
- Multicast üzenet hatására (kivételek: Packet Too Big, Parameter Problem)
- Ha egy csomagot Adatkapcsolati rétegbeli multicast üzenetként küldtek
- Ha egy csomagot Adatkapcsolati rétegbeli broadcast üzenetként küldtek

- Olyan csomag hatására, amelynek a forrás címe nem azonosít egyértelműen egy csomópontot

Ezen felsorolás utolsó két pontja minden esetben elsőbbséget élvez minden egyéb hasonló megkötéssel szemben.

A következőkben az egyes hiba és tájékoztató jellegű üzenetek szerkezetét mutatom be.

## *3.1 Hibaüzenetek*

### 3.1.1 Cél Nem Elérhető (Destination Unreachable)

| Típus | Kód | Elenőrző összeg |
|---|---|---|
| Nem használt | | |
| A kiváltó csomagból a lehető legtöbb úgy, hogy lehetőleg az ICMPv6-os fejlécek nélkül meghaladja a minimális MTU-t | | |

Az IPv6-os fejléc cél címe a kiváltó csomagban szereplő forrás cím lesz. Az ICMPv6-os mezők a következőképpen alakulnak:

Típus: 1

Kód: 0 - Nincs út a célig
1 - A céllal folytatott kommunikáció adminisztratívan tiltott
2 - Forrás cím hatáskörén kívül
3 - Cím nem elérhető
4 - Port nem elérhető
5 - Cél cím bejövő/kimenő policyt sért
6 - Célhoz vezető út visszautasítva

A Nem használt mezőt mindig nullákkal kell feltölteni, és használata nem engedélyezett egyik kód számára sem.

A Cél Nem Elérhető üzenetet router, vagy a forrás állíthatja elő, ha a cél nem forgalmi akadály miatt nem elérető. Ha a továbbítás azért nem sikerült, mert valamelyik köztes csomópont routing táblájában nincs bejegyzés a célra vonatkozóan, akkor a nullás kódot kell beállítani (ez csak akkor fordulhat elő, ha az adott csomópontnak nincs alapértelmezett átjáró beállítva). Ha például egy tűzfal miatt nem sikeres a küldés, akkor az 1-es értéket kell beállítani. Ha például egy Link-Local forrás címről szeretnénk egy globális hatáskörű címre küldeni, akkor 3-mas hibakódú üzenetet kapnánk.

Biztonsági okokból ezt a típusú ICMP hibaüzenetet le lehet tiltani. Amennyiben egy csomópont Cél Nem elérhető hibaüzenetet kap, figyelmeztetnie kell a felsőbb rétegbeli protokollt is, amennyiben meg tudja határozni, hogy melyik az a protokoll.

### 3.1.2 Csomag Túl Nagy (Packet Too Big)

Az üzenet szerkezete a következő:

| Típus | Kód | Elenőrző összeg |
|---|---|---|
| MTU | | |
| A kiváltó csomagból a lehető legtöbb úgy, hogy lehetőleg az ICMPv6-os fejlécek nélkül meghaladja a minimális MTU-t | | |

A cél címet ebben a esetben is a kiváltó csomagból fogjuk megtudni, a többi mező értékei pedig a következők szerint alakulnak:

Típus: 2

Kód: Ezt a mezőt küldéskor minden esetben nullával kell inicializálni, és figyelmen kívül kell hagyni fogadáskor.

MTU A következő linken érvényben lévő MTU

Ezt az üzenetet egy router küldheti egy olyan csomagra adott válaszként, amit azért nem tud tovább küldeni, mert a csomag mérete meghaladja az adott linken egyszerre átvihető bájtok számát. Amennyiben egy csomópont Csomag Túl Nagy hibaüzenetet kap, figyelmeztetnie kell a felsőbb rétegbeli protokollt is, amennyiben meg tudja határozni, hogy melyik az a protokoll.

### 3.1.3 Időtúllépés (Time Exceeded)

Az üzenet felépítését tekintve megegyezik a Cél Nem Elérhető típusú üzenettel, ezért azt nem részletezem. Az egyes mezők értékei a következők:

Típus: 3

Kód: 0 - Átvitel közben meghaladtuk a maximális ugrásszámot

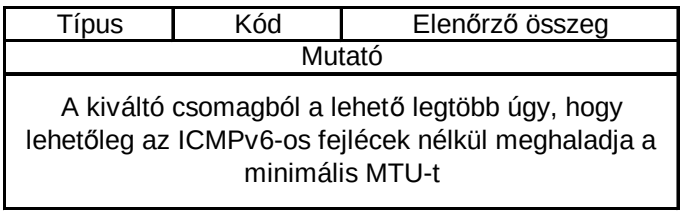
1 - Töredék összeállítási ideje lejárt

A Nem használt mezőt mindig nullákkal kell feltölteni, és használata nem engedélyezett egyik kód számára sem. Amennyiben egy router nullás Hop-Limittel rendelkező csomagot fogad, vagy a router a Hop-Limit értékét nullára csökkenti, abban az esetben a csomagot el kell

dobni, és egy ICMP hibaüzenetben jelezni kell azt. Ilyen eset több okból is előfordulhat, például, mert nem megfelelő értékkel inicializáltuk a Hop-Limit mezőt, vagy mert a csomagunk egy forgalomirányítási hurokban bolyongott mindaddig, amíg a mező értéke nullára nem csökkent. Amennyiben egy csomópont Időtúllépés hibaüzenetet kap, figyelmeztetnie kell a felsőbb rétegbeli protokollt is, amennyiben meg tudja határozni, hogy melyik az a protokoll.

### 3.1.4 Parameter Problem

A következőképpen néz ki:




Az IPv6-os csomag cél címét ebben a esetben is a kiváltó csomagból fogjuk megtudni, a többi mező értékei pedig a következők szerint alakulnak:

Típus: 4

Kód: 0 - Hibás fejléc mező

1 - Nem ismert következő fejléc típus

2 - Nem ismert IPv6-os opció

Amennyiben egy csomópont egy csomag feldolgozása közben olyan értékkel találkozik, amit nem ismer fel, akkor azt a csomagot el kell dobnia, és lehetősége van a feladót erről egy ICMPv6-os üzenetben tájékoztatni. Mint a kódok neveiből kitűnik, az 1-es és 2-es kód informatívabb változata a nullás kódnak. A Mutató azonosítja az eredeti csomag fejlécének a hibás bájtját. Amennyiben egy csomópont Parameter Problem hibaüzenetet kap, figyelmeztetnie kell a felsőbb rétegbeli protokollt is, amennyiben meg tudja határozni, hogy melyik az a protokoll.

## *3.2 Tájékoztató jellegű üzenetek*

### 3.2.1 Visszhang Kérés (Echo Request)

Az Echo Request üzenet a következő módon épül fel:

| Típus | Kód | Elenőrző összeg |
|---|---|---|
| Azonosító | | Sorszám |
| Adat… | | |

Az IPv6-os fejléc cél cím mezejében ilyenkor bármilyen, érvényes IPv6-os cím szerepelhet.
Az ICMPv6-os üzenet mezői a következők:

Típus: 128

Kód: 0

Azonosító: A visszhang kérések és azokra adott válaszok összepárosítására szolgál. Értéke lehet nulla.

Sorszám A sorszám mező funkcióját tekintve megegyezik az azonosító mezővel. Értéke szintén lehet nulla.

Adat Nulla vagy több bájtnyi tetszőleges adat.

Minden csomópontnak képesnek kell lenni Echo üzenetek küldésére és fogadására.

### 3.2.2 Visszhang Válasz (Echo Reply)

Felépítését tekintve azonos a Visszhang Kérés üzenettel, és a mezők értékei is az elvárásainknak megfelelően alakulnak. Az IPv6-os fejléc cél címe a kiváltó üzenet forrás címe lesz. A Visszhang Válasz üzenet mezőinek értékei a következők:

Típus: 129

Kód: 0

Azonosító: A kiváltó üzenet hasonló mezőjének az étéke

Sorszám A kiváltó üzenet hasonló mezőjének az étéke

Adat A kiváltó üzenet hasonló mezőjének az étéke

Lehetőség van az ICMP csomagokat és azok tartalmát az Authentication Header és az ESP Header segítségével védeni. Ez azért is fontos, mert az ICMP üzenetek különböző támadásokra is lehetőséget adnak. Ilyen támadás lehet például, amikor a feladó szándékosan más forrás vagy cél címet helyez el a fejlécben, mint ami a tényleges cím. Mivel az ICMP csomagot a feldolgozás során átadhatják valamelyik felsőbb rétegbeli protokollnak, ezáltal azok is támadhatóvá válnak. A Neighbor Discoveryhez kapcsolódóan még bemutatok néhány ICMP üzenet fajtát.

# 4 Szomszédság felderítés (Neighbor Discovery)

IPv6-os környezetben a Neighbor Discovery protokoll felel meg az IPv4-es ARP, ICMP Router Discovery és az ICMP Redirect funkcióknak. Ebben a fejezetben tehát ennek az IPv6 szolgáltatásnak az alapjait ismertetem. Korábban már említettem néhány ilyen szolgáltatást, mint például a Router Discoveryt, vagy a Neighbor Solicitationt. A Szomszédság Felderítés a következő problémákra nyújt megoldást:

- Router Discovery: Hogyan tudja egy hoszt megállapítani az adott linken a router helyét.
- Prefix Discovery: Hogyan tudja kideríteni a címek prefixeit, ami alapján meg tudja mondani, hogy melyik cél címmel van egy linken, és melyiket tudja csak a routeren keresztül elérni.
- Parameter Discovery: Hogyan tudja megállapítani a különböző link paramétereket (például az MTU) vagy az internet paramétereket (például Hop Limit) a kimenő csomagok számára.
- Address Autoconfiguration: Hogyan tudja a hoszt megállapítani az állapotmentes autókonfigurációhoz szükséges információkat.
- Address Resolution: Hogyan tudja a hoszt megállapítani a vele egy linken lévő cél IP címéből annak fizikai címét.
- Next Hop Determination: Hogyan tudja a cél címet egy szomszédja IP címével összerendelni annak érdekében, hogy meg tudja határozni, hogy kinek kell továbbküldenie a csomagot. A szomszéd egyaránt lehet egy router és maga a cél is.
- Neighbor Unreachability Detection: Hogyan tudja észlelni a csomópont, hogy valamelyik szomszédja nem elérhető.
- Duplicate Address Detection: Hogyan tudja eldönteni a csomópont, hogy a használni kívánt cím egyedi-e a hálózaton.
- Redirect: Hogyan tud egy router értesíteni egy másik csomópontot egy jobb, célhoz vezető útról.

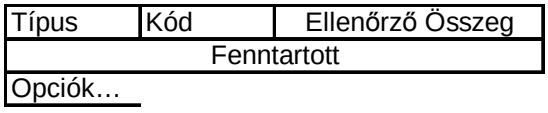
A Szomszédság Felderítés öt különböző ICMP üzenet formát definiál annak érdekében, hogy elősegítse ezen funkciók megvalósíthatóságát:

A Router Solicitation üzeneteket az interfészek használják annak érdekében, hogy a routereket azonnali Router Advertisement üzenet előállítására késztessék. A Router Advertisement üzeneteket a routerek küldik szét meghatározott időközönként vagy egy Router Solicitation üzenet eredményeként. Az üzenetben az adott linkre és egyéb, internetre vonatkozó információk is szerepelnek. A Neighbor Solicitationt azért küldik a csomópontok, hogy a szomszédjaik fizikai címét megállapíthassák, vagy azért, hogy ellenőrizzék, hogy az általuk használni kívánt cím nem foglalt-e. Ilyen üzenetet küldenek akkor is, ha arra kíváncsiak, hogy egy szomszédjuk még mindig elérhető-e az általuk ismert címen. A Neighbor Solicitation üzenetre küldött válasz a Neighbor Advertisement. Ezt az üzenetet kérés nélkül is szétküldhetik a hosztok annak érdekében, hogy tájékoztassák a szomszédaikat a címükben bekövetkezett változásról. Redirect üzenetet akkor küldenek, amikor arról tájékoztatják valamelyik szomszédukat, hogy az adott célhoz jobb út is létezik az általuk használttól.

A routerek elég sűrűn küldenek információkat ahhoz, hogy a hosztok néhány perc alatt megtanulják az őket körülvevő topológiát, de nem elég sűrűn ahhoz, hogy észlelhessék, ha egy forgalomirányító elérhetetlen. Ezt a Neighbor Unreachability Detection segítségével tudják érzékelni. A hosztok a Router Advertisement üzenetekből felépítenek, és karban tartanak egy listát, amibe azokat a prefixeket gyűjtik össze, amelyeket a routerek hirdetnek. Így fogják tudni, hogy melyik cím van ugyanazon a linken, és melyiket nem érik el közvetlenül. Vannak azonban olyan címek is, amiknek a prefixeit a forgalomirányítók nem hirdetik, pedig közvetlenül elérhetők a hoszt számára. Ekkor a hoszt az erre a címre küldendő csomagokat a routernek fogja küldeni, az pedig egy Redirection üzenetben tájékoztatni fogja a forrást, hogy van jobb út is a célhoz, mégpedig a közvetlen elérésű link-local cím.

Természetesen az itt bemutatott eszközök mindegyike védhető az AH és ESP fejlécek segítségével.

## *4.1 Router Solicitation üzenet*



Az IPv6-os fejléc forrás cím mezője egy IPv6-os cím, vagy a nem meghatározott cím, amennyiben még nincs hozzárendelve cím az interfészhez. A cél cím tipikusan a „minden router multicast” cím, a Hop Limit pedig 255.

Az ICMPv6-os üzenet mezői a következők:

Típus: 133

Kód: 0

Az ellenőrző összeg és a Fenntartott mező értéke és jelentése megegyezik a korábban elmondottakkal. Érvényes opció lehet a forrás csomópont fizikai címe, azonban ha a forrás cím a nem meghatározott cím, akkor nem kell megadni

## *4.2 Router Advertisement üzenet*

<table>
<tr><td>Típus</td><td colspan="3">Kód</td><td>Ellenőrző összeg</td></tr>
<tr><td>Akt.Hop Limit</td><td>M</td><td>O</td><td>Fenntartott</td><td>Router Élettartama</td></tr>
<tr><td colspan="5">Elérhetőség ideje</td></tr>
<tr><td colspan="5">Újraküldési időköz</td></tr>
<tr><td>Opciók</td><td colspan="4"></td></tr>
</table>

Az IPv6-os fejléc forrás mezejében egy Link-Local címnek kell állni, a cél cím pedig vagy a kiváltó csomópont címe, vagy egy „all-nodes” multicast cím. A Hop Limit ebben az esetben is 255-re lesz beállítva.

Az ICMPv6-os üzenet mezői a következők:

Típus: 133

Kód: 0

Akt. Hop Limit: 8 bites előjel nélküli egész, az alapértelmezett Hop Limit értékét adja meg. Nullás értéke jelzi, hogy nincs ilyen érték meghatározva.

M 1 bites, 1-es érétke jelzi, hogy van-e lehetőség DHCP segítségével IPv6-os címet kapni. Amennyiben az érték be van állítva, az O bit redundáns információt hordoz, és feldolgozásnál figyelmen kívül lehet hagyni.

O 1 bites, 1-es értéke jelzi, hogy a címen kívül más információk is elérhetők DHCP- keresztül. Amennyiben sem az M, sem az O bit nincs beállítva, az azt jelenti, hogy a DHCP nem elérhető.

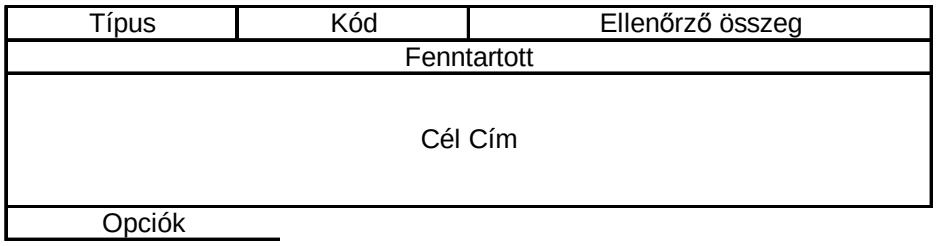
Router Élettartama: 16 bites előjel nélküli egész, az alapértelmezett router élettartamát adja meg másodpercekben. Nullás értéke jelzi, hogy az adott router nem alapértelmezett routerként funkcionál.

Elérhetőség Ideje: 32 bites egész, értéke ezred másodpercekben mutatja azt az időt, amin belül elérhetőnek kell feltételezni az adott szomszédot.

Újraküldési időköz: 32 bites előjel nélküli egész, értéke milliszekundumban mutatja, hogy milyen időközönként lehet újraküldeni egy Szomszédság Kérése üzenetet.

A lehetséges opciók között szerepel az MTU, a prefix hosszra vonatkozó információk, és a forrás fizikai címe is.

## 4.3 Neighbor Solicitation üzenet




Az IPv6-os fejléc forrás cím mezője egy IPv6-os cím, vagy a nem meghatározott cím, amennyiben még nincs hozzárendelve cím az interfészhez. A cél cím tipikusan a cél cím, vagy annak a multicast címe, a Hop Limit pedig 255.

Az ICMPv6-os üzenet mezői a következők:

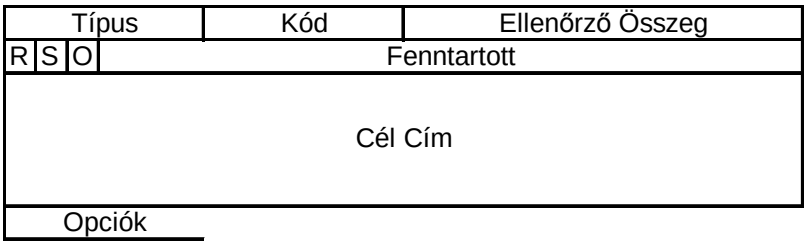
Típus: 135

Kód: 0

Cél cím: a kérés céljának a címe, ami nem lehet multicast cím.

Lehetséges opcióként a forrás fizikai címe szerepelhet a csomagban, ebben az esetben viszont nem lehet forrásként a nem meghatározott IPv6-os címet adni

## 4.4 Neighbor Advertisement üzenet



Az IPv6-os fejléc forrás mezője minden esetben a küldő címe, a cél cím pedig függ a küldés miértjétől. Amennyiben egy Neighbor Solicitation üzenetre küldjük válaszként, akkor a kiváltó üzenet forrás címe (amennyiben az a nem meghatározott cím, akkor az „all-node” multicast cím), ha viszont csak a címünkben bekövetkezett változást akarjuk hirdetni, akkor az „all-node” multicast cím. A Hop-Limit ebben az esetben is 255.

Az ICMPv6-os üzenet mezői a következők:

Típus: 136

Kód: 0

R 1-es értéke mutatja, hogy a küldő egy forgalomirányító volt.

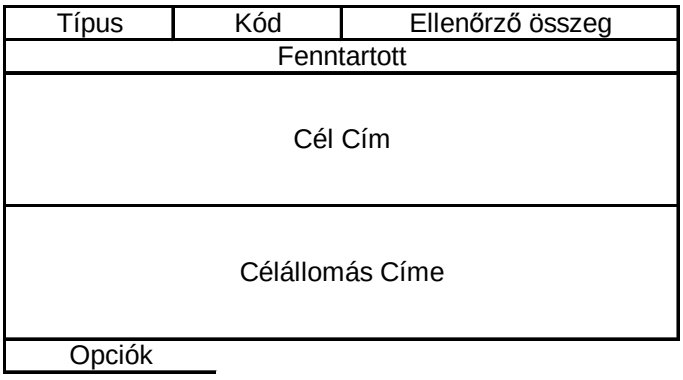
S 1-es értéke mutatja, hogy az üzenet egy kérésre lett küldve.

O 1-es értéke jelzi, hogy az üzenet hatására a tárolt fizikai címhez tartozó bejegyzés megváltozhat

Cél cím: Kért hirdetések esetén a kérelmező címe, nem kért hirdetés esetén pedig annak a címe, akinek a fizikai címe megváltozott. Egyetlen esetben sem lehet multicast cím.

Lehetséges opcióként a cél fizikai címét lehet elhelyezni az üzenetben. Amennyiben multicast hirdetésről van szó ezt az opciót lehetőleg alkalmazni is kell annak érdekében, hogy elkerüljük a végtelen hirdetési hurkok kialakulását.

## *4.5 Redirect üzenet*




Az IPv6-os fejléc forrás mezője minden esetben a küldő címe, a cél cím pedig az üzenetet kiváltó csomag feladójának a címe. A Hop-Limit szintén 255.

Az ICMPv6-os üzenet mezői a következők:

| | |
|---|---|
| Típus: | 137 |
| Kód: | 0 |
| Cél cím | Annak a csomópontnak a címe, amelyik közelebb van a csomag küldőjéhez. Ha a cél cím az eredeti csomag cél címe, akkor a csomag egy szomszédnak lett címezve, akit pedig közvetlen elér a küldő, tehát nincs szüksége routerre. Ha a két cím eltér, akkor az egy jobb utat jelöl. |
| Célállomás címe | A célállomás címe, amelyhez a célon keresztül jobb út vezet. |

Opcióként szerepelhet a cél fizikai címe, valamint az átirányított fejlécből annyi, amennyi csak lehet úgy, hogy az átirányított csomag túllépné a minimális MTU méretet.

Könnyen belátható, hogy a most bemutatott üzenetek képesek úgy irányítani a csomagokat, hogy azok egy nem kívánt helyre folyjanak, DoS típusú támadást megvalósítva ez által. Ezen kívül használhatók még router és cím spoofingra is.

# 5 IPSec

Az IPv6 szabványosítása során központi kérdés volt a biztonság, ugyanis az „internetes bűnözés” egyre népszerűbb. Mivel a hálózati kommunikáció egy része kódolatlanul halad a végpontok között, ezért sokszor különösebb erőfeszítést sem kell tenni egy-egy felhasználónév/jelszó páros megszerzésére (HTTP, POP3, stb.). De titkosítás és hitelesítés nélkül nem csak a felhasználói adatainkra utazóknak van könnyű dolguk, hanem azoknak a támadóknak is, akik szándékosan próbálják lebénítani a hálózatunkat oly módon, hogy eszközeinket túlterhelik, Denial of Service típusú támadást megvalósítva ezzel. Mivel a támadások is sokrétűek lehetnek, az ellenük való védekezésnek is összetettnek kell lenni. Ennek az egyik alapvető pillére pedig az IPSec. Ebben a fejezetben a teljesség igénye nélkül, de kellő alapossággal igyekszem bemutatni az IPSec szolgáltatásait, az általa nyújtott megoldásokat.

Az IPSec egy határvonalat képez a védett és a nem védett terület között, és ezen a határvonalon átlépő csomagokat vagy átengedi az AH és ESP-nek megfelelően, vagy megakadályozza az áthaladásukat. Az IPSec tehát az Authentication Headerre és az Encapsulation Security Payload Headerre alapozza a szolgáltatásait, valamint szerves részét képezi még az Internet Key Exchange (IKE), ami a kapcsolat paramétereinek megvitatására és kulcscserére szolgál. Ezek segítségével épül fel egy Biztonságos Kapcsolat (SA).

Az SA-k képezik az IPSec alapját, ugyanis a biztonságos kommunikáció biztonságos kapcsolatot követel. Egy ilyen kapcsolat biztonságáért vagy az AH, vagy az ESP felel, de egyszerre mind a kettő nem használható. Ha azonban valamilyen oknál fogva két végpont között mégis szükség van mindkét szolgáltatás igénybe vételére, akkor két külön SA-nak kell kiépülni. Egy tipikus, kétirányú kommunikáció során természetesen mindkét félnél ki kell, hogy épüljön egy-egy SA, ezek pedig az IKE segítségével összepárosíthatók.

Amennyiben unicast kommunikációt folytatnak a partnerek, az SPI önmagában is elegendő az SA azonosítására. Azonban multicast kommunikáció során már szükséges egy koordinátor is. Ez lesz a Group Controller, vagy Kulcs Szerver, ami önhatalmúan osztja ki ezeket a csoportos biztonsági kapcsolatra (GSA) vonatkozó SPI értékeket, amelyeket az egyes végpontok kulcskezelő mechanizmusai nem utasíthatnak el, vagy nem bírálhatnak felül (lehetséges, hogy ugyanazon SPI érték egy unicast és egy multicast SA-t is azonosít). Ahogy az Authentication Headernél is leírtam, az SA-kat egy adatbázisban tárolják (SAD). Ez az adatbázis minden SA-

hoz tartalmaz egy bejegyzést, ami azt mondja meg, milyen módban használják az adott SA-t. Egy kapcsolat ugyanis transzport és tunnel módban is képes működni. Nézzük most ezeket külön-külön.

## 5.1 Transzport mód

Tipikusan két hoszt között használják, végponttól végpontig terjedő biztonság nyújtására. Ebben a módban a biztonsági protokoll fejléce az IP fejléc és a kiterjesztett fejlécek után szerepelhet, de előfordulhat még a Destination Options Header előtt vagy után, valamint elő kell fordulnia a felsőbb rétegbeli protokoll fejléce előtt. ESP esetén csak a felsőbb rétegbeli protokoll védett, az ESP fejlécet megelőző IP fejlécek és opciók nem. AH esetén ez a védelem kiterjed az IP fejléc bizonyos mezőire és egyes opciókra is. A transzport mód előnye, hogy a két végpont közti hálózatrészen különböző speciális feldolgozást végezhetünk (QoS), valamint kevés többletinformációt kell a kommunikáció során átvinnünk.

## 5.2 Tunnel mód

Ebben a módban az eredeti csomagunk becsomagolódik egy újabb IPv6-os csomagba, és a biztonsági fejléc a két IP fejléc közé kerül, Ennek köszönhetően az eredeti csomagunk mind AH, mind ESP esetén teljesen védett. AH esetén ez igaz az új IPv6-os fejléc bizonyos mezőire is. Ez a mód azért fontos, mert itt nem az egyes hosztok, hanem a routerek végzik a titkosítást, és az egyéb műveleteket, valamint mert így akár egy egész alhálózat forgalmát is át lehet ugyanazon az adott SA-n vinni (mivel nem minden hoszt képes IPsec alapú kommunikációra, az ők forgalmuk is biztosíthatóvá válik a tunnel mód használatával.).

Az IPSec szabványosítása során definiáltak 3 adatbázist, amit minden implementációnak használni kell a minimális kompatibilitás és a menedzselhetőség érdekében (ez persze nem jelenti azt, hogy csak ezeket tartalmazhatják). Ez a három pedig a következő:

- Security Policy Database
- Security Association Database
- Peer Authorization Database

## 5.3 Security Policy Database (SPD)

Egy rendezett adatbázis, ami leírja, hogy milyen szolgáltatásokat és milyen módon vehetnek igénybe az IP datagramok. Ezt minden adatforgalom esetén használják (bejövő és kimenő egyaránt), beleértve azokat a csomagokat is, amelyek úgy lépik át az IPSec határvonalát, hogy nincsenek általa védve. Az IPSec implementációknak tartalmazni kell legalább egy ilyen adatbázist, de lehetőség van több SPD használatára is, ekkor azonban meg kell tudni mondanunk, hogy melyikkel kell dolgoznunk. Több interfész esetén nem szükséges külön SPD-ket karban tartani minden interfészhez, de szükség esetén természetesen megtehetjük, ha fennállnak az előbbi feltételek. A rendezettséget azért kell megkövetelnünk, mert a bejegyzései átfedhetik egymást a különböző szelektorok értékei miatt. Mivel joker karaktereket (OPAQUE, ANY; lásd később) is adhatunk a szelektor értékeinek, és a típusoknak sincs hierarchiájuk, ezért nem lehetséges egy általánosan helyes, kanonikus rendezést előírni.

Egy datagramal (legyen az bejövő, vagy kimenő) feldolgozása során 3 dolog történhet:

- A csomag nem lépheti át az IPSec határát, ezért eldobják (DISCARD)
- A csomagot nem védi IPSec mechanizmus, de átlépheti a határvonalat (BYPASS)
- A csomag az IPSec által védve van, és át is lépheti a határvonalat (PROTECT); Ebben az esetben az SPD-nek meg kell tudnia határozni a biztonsági protokollt, annak módját, a biztonsági opciókat és a használt kriptográfiai algoritmust is

Az adatbázis ebből kifolyólag 3 logikai részre tagolható. Az első része (SPD-S) azokat a bejegyzéseket tartalmazza, amelyek IPSec által védett kommunikációhoz tartoznak. Az SPD-O azokat a kimenő adatfolyamokat azonosítja, amelyeket vagy blokkolni kell, vagy átengedni (BYPASS). A harmadik rész az SPD-I nevet kapta, mely az SPD-O-val analóg módon (bypass vagy discard) kezel bejövő adatfolyamokat. A gyorsítótárazás érdekében ezek a részek nem fedhetik át egymást.

Mint már említettem, definiáltak joker karaktereket is a szelektorok megadásához. Ezek a következők:

- OPAQUE: azt jelöli, hogy az adott mező nincs jelen, vagy nem létezik, vagy titkosítva van az értéke; akkor van jelentőssége, ha különbséget kell tenni az érvényes érték és az érték hiánya között.
- ANY: Olyan joker karakter, ami bármilyen értékre illeszkedik, beleértve a nem létező, vagy nem látható értékeket is.

## 5.4 A szelektorok

Egy SA finom szemcsézettségű vagy durva szemcsézettségű lehet annak megfelelően, hogy a szelektor milyen forgalmat definiál az SA számára. Például ha egyetlen SA-n keresztül folyik az összes kommunikáció, az durva szemcsézettségre utal. A másik végletet az jelenti, amikor minden különböző adatfolyamhoz külön SA épül ki (finom szemcsézettség).A kettő között természetesen tetszőleges átmenet lehet, például felsőbb rétegbeli protokollonként egy SA, vagy célonként különböző SA, és így tovább. Fontos még megjegyezni, hogy a forrás és a cél cím is lehet IPv4 vagy IPv6-os cím, de nem lehet ezeknek a keveréke (ugyanazt a típust kell mindkettőnek használni). Szemcsézettségtől függetlenül minden IPSec implementációnak támogatni kell a következő szelektorokat:

- Távoli IP cím(ek): IP címtartományok egy listája
- Helyi IP cím(ek): IP címtartományok egy listája
- Felsőbb Rétegbeli Protokoll: Egy egyedi protokoll azonosító, ANY vagy OPAQUE. Felsőbb Rétegbeli Protokollnak nevezünk bármilyen protokollt, ami valamilyen kiterjesztett IP fejléc után következik. Annak érdekében, hogy ezt könnyebben meghatározhassuk, bizonyos fejléceket átugorhatunk. Ezek a Hop-by-Hop, Routing, Fragmentation és Destination Fejlécek. Az AH és az ESP nem ugorható át. A szelektorok szemszögéből tehát az AH és az ESP felsőbb rétegbeli protokollok.
- Számos további szelektor (portszám, MH type, ICMP, stb.) a felsőbb rétegbeli protokolltól függ, ezért ezeket nem tárgyalom

A Név szelektor nem az eddigiekhez hasonló szelektor, értékét ugyanis nem valamilyen csomagból szerezzük, hanem egy helyi vagy távoli cím címkéjeként használjuk.

## 5.5 Az SPD bejegyzések felépítése

Egy SPD bejegyzés a következőket tartalmazza:

- Név: azonosítók(Nevek) listája, nem kötelező
- PFP flag: Populate From Packet; adatfolyamonként egy szelektor; jelzi, hogy az adott szelektor értékét a csomagból származtatják-e
- Legalább 1 Szelektorhalmazt. Amely leírja, hogy milyen körülmények között kell az adott bejegyzést alkalmazni. Minden ilyen halmaz tartalmaz egy helyi, egy

távoli címet, egy felsőbb rétegbeli protokollt, és az annak megfelelő további adatokat

- Feldolgozásra vonatkozó információk: PROTECT, BYPASS vagy DISCARD; SPD bejegyzésenként 1 darab (nem szelektor halmazonként); PROTECT esetén további információkat (IPSec módja, protokoll típusa, titkosító algoritmus, stb.)

## 5.6 Security Association Database (SAD)

Minden bejegyzés egy SA-t azonosít, a következő bejegyzéseknek mindig jelen kell lenni:

- SPI érték
- Sorszám számláló
- Egy flag, ami jelzi, hogy a Sorszám számláló túlcsordulhat-e
- Anti-Replay Window
- AH authentikációs algoritmus, kulcs, stb; csak akkor kell, ha támogatja az AH-t
- ESP titkosító algoritmus, kulcs, mód, stb; ha a kombinált mód támogatott, akkor ezek nem kellenek
- ESP integritás ellenőrző algoritmus, kulcs, stb; ha az integritás ellenőrzés nem támogatott, akkor nem kellenek; ha a kombinált mód támogatott, akkor nem kellenek
- ESP kombinált módú algoritmus, kulcs, stb. Csak akkor alkalmazandó, ha a titkosítás és az integritás ellenőrzés is használva van.
- SA élettartama
- IPSec protokoll módja (tunnel, transport)
- Stateful fragment checking flag
- Bypass DF bit (igaz/hamis)
- DSCP értéke
- Bypass DSCP (igaz/hamis)
- Út MTU-ja
- Tunnel módú IP fejléc forrás és cél címe

## 5.7 Peer Authorization Database (PAD)

A PAD teremti meg a kapcsolatot az SPD és a különböző menedzselő protokollok (pl. IKE) között. A következő funkciókat testesíti meg:

- Azonosítja azokat a felhasználókat, vagy felhasználók azon csoportját, akik kommunikálhatnak ezzel az IPSec entitással
- Meghatározza a peerek hitelesítésének protokollját és módját
- Hitelesítési adatokat szolgáltat a peerek számára
- Különböző megszorításokat a gyerek SA létrehozására vonatkozóan
- Átjáró információk (DNS, IP címek, stb.)

A PAD egy olyan rendezett adatbázis, ahol a rendezettséget az adminisztrátor definiálja

## *5.8 Kulcskezelés (IKE)*

Az AH és ESP működéséhez szükség van egy-egy titkos kulcsra. Ezeket a kulcsokat manuálisan, és automatikusan is elő lehet állítani. Manuális kulcskonfiguráció esetén a rendszeradminisztrátor minden egyes résztvevő hoszton kézzel konfigurálja a hoszt saját és a kommunikáló partnerei kulcsait. Ez nyilvánvalóan csak kisszámú hoszt és statikus konfiguráció esetén tartható kézben. A manuális kulcskezelés következményeként elvész a replay támadások elleni IPSec védelem (nem lehet új SA-t létrehozni). Az automatizált kulcskezelés ezzel ellentétben lehetővé teszi az új SA-k létrehozását, hiszen képes kulcsot generálni, továbbá azt „terjeszteni” is.

Az automatikus kulcskezelésre az IKE protokollt használják, ami az SA-k kezelését is biztosítja. Segítségével egyszerűsödik az SA-k felépítése és a kulcs csere a kommunikáló felek között. A kulcsok segítségével csak az üzenet küldője és fogadója férhet hozzá az üzenethez, az IPSec azonban megkívánja, hogy a kulcsokat gyakran váltogassuk. Az IKE ezt a kulcsváltási folyamatot is kezeli, ahol a felhasználó szabhatja meg a kulcs erősségét, és hogy milyen gyakran történjen kulcscsere (idő vagy adatforgalom korlát megadásával). Az IKE az egyik legjobban kidolgozott, és az egyik legelterjedtebb kulcscserélő protokoll a hálózati rétegen. Az IKE egy másik szabványos protokollt használ fel, az ISAKMP-t (Internet Security and Key Management Protocol), amely nem írja elő a kulcscsere algoritmusát, hanem olyan üzenetkészletet tartalmaz, amellyel többféle kulcscsere is elvégezhető. A kulcscseréhez mindkét kommunikáló fél létrehoz egy-egy nyilvános-privát kulcspárt, és a nyilvános kulcsot elküldi a kommunikációs partnernek, majd a csomag titkosításhoz használt titkos kulcsot ezzel a nyilvános kulccsal titkosítva továbbítják. Megfelelő hitelesítés használatával ez elég biztonságos eljárás lehet, főleg ha a kulcsokat rendszeresen cserélik. A hitelesítéshez az úgynevezett PKI-tanúsítványok használhatóak.

Az IKE kétfázisú folyamatként működik. Az első fázis felépíti az aktuális IKE SA-kat, a második fázis pedig felépíti a biztonságos adatátviteli csatornákat, azaz az IPSec SA-kat. A protokoll négy módot használ: a fő módot (Main Mode), az agresszív módot(Agressive Mode), a gyors módot (Quick Mode) és az új csoport módot (New Group Mode). A fő mód és az agresszív mód az 1. fázisban használatosak, míg a gyors mód egy második fázisú üzenetcsere. Az új csoport mód egy kicsit eltér ezektől: se nem első fázisú, se nem második fázisú üzenetcsere, viszont csak akkor hajtódhat végre, ha már létezik egy felépített ISAKMP SA. Arra használjuk, hogy egy új csoportot egyeztessünk a későbbi Diffie-Hellman cserékhez. Az 1. fázis során az IKE a következő attribútumokat egyezteti (és később ezeket fogja használni):

- titkosító algoritmus
- hash függvény
- hitelesítési eljárás
- Diffie-Hellman csoport

Az első fázis végén három kulcsot generálunk: egyet a titkosításhoz, egyet a hitelesítéshez, és egyet további kulcsok generálásához. A kiszámításuk az ISAKMP SA-hoz meghatározott hash függvény felhasználásával történik, és függ a kiválasztott hitelesítési módtól is.

### 5.8.1 Az első fázis

A fő mód hat üzenetet tartalmaz, ebből az első kettő az SA egyeztetéséhez, a második kettő a Diffie-Hellman nyilvános értékek kicseréléséhez, míg az utolsó kettő a nyilvános értékek hitelesítéséhez szükséges. Az agresszív mód csak három üzenetből áll, és az ISAKMP Aggressive Exchange módjának a megvalósítása. A fenti két módban a hitelesítéshez választott eljárás befolyásolja az üzenetek tartalmát és a viszonykulcs-generáló eljárást.

### 5.8.2 A második fázis

A második fázis során váltott üzeneteket az első fázis során egyeztetett értékek segítségével védjük. Az üzenetek hitelességét az ISAKMP fejrész után beillesztett HASH adatrésszel biztosítjuk, míg a bizalmasságot az üzenet összes adatrészének a titkosításával garantáljuk. A gyors módot az SA-k egyeztetéséhez használjuk, az általa tartalmazott üzenetváltások szerepe a következő:

- Egy IPSec paraméter készlet egyeztetése (SA kötegek)

- Véletlen számok cseréje egy új viszonykulcs generálásához, amely viszonykulcsot az első fázis során generált titokból származtatunk.
- Az SA-köteg által védendő adatfolyam azonosítása szelektorok használatával (itt általában a felek IP címét használjuk fel).

Mint az elején már említettem, ezen leírás közel sem teljes, és minden igényt kielégítő. Mivel a dolgozat fő témája nem az IPSec, ezért igyekeztem csak az általam legfontosabbnak vélt pontokat kiemelni a témához tartozó szabványokból, szakirodalmakból.

# 6 DNS

Az IP címek méretváltozása miatt az ICMP mellett a DNS-nek is át kell esni egy „ráncfelvarráson”. Mivel az IPv6 nem igényli az eddig működő DNS rendszer teljes lecserélését, ezért csak a lényeges újításokra vagy módosításokra szorítkozom. A teljes rendszer lecserélése nagyon nehéz feladat lenne, hiszen a jelenleg létező alkalmazások döntő hányada 32 bites címet vár egy DNS kérés eredményeként. A következő változások azonban megtartják a kompatibilitást, és képessé teszik a jelenlegi rendszert az IPv6-os címek kezelésére:

- Egy új erőforrás rekord bevezetése az IPv6-os címeknek
- Olyan domain definiálása, amely támogatja címen alapuló keresést
- A létező lekérdezéseket újra definiálták, hogy támogassák az IPv4-es és IPv6-os címek feldolgozását is. (Ha keresnek A-ra, akkor AAAA rekordokra is keressenek)

Ezeket a változásokat úgy tervezték meg, hogy a létező szoftverekkel kompatibilisek legyenek, és hogy a lekérdezéshez használt IP protokoll verziószáma ne legyen befolyással az eredményre (IPv4-el is le lehet kérdezni IPv6-os rekordot, és fordítva.)

Az Új rekord

Az IPv6-os címek tárolására szükség van egy rekordra. Minden olyan hoszt, amelynek egynél több IPv6-os címe van, egynél több ilyen rekorddal kell, hogy rendelkezzen. Ezt a rekordot AAAA rekordnak nevezték el, és egyetlen 128 bites cím tárolására használják. A cím, mint minden IPv6-os cím, a már bemutatott formában jelenhet meg az adatbázisban. A tárolás során természetesen hálózati bájtsorrendet használnak (a legmagasabb helyi értékű bit a legelső). Amennyiben egy domain névhez több AAAA rekord is hozzá van rendelve, a lekérdezés eredményeként az összeset megkapjuk.

Az Új domain

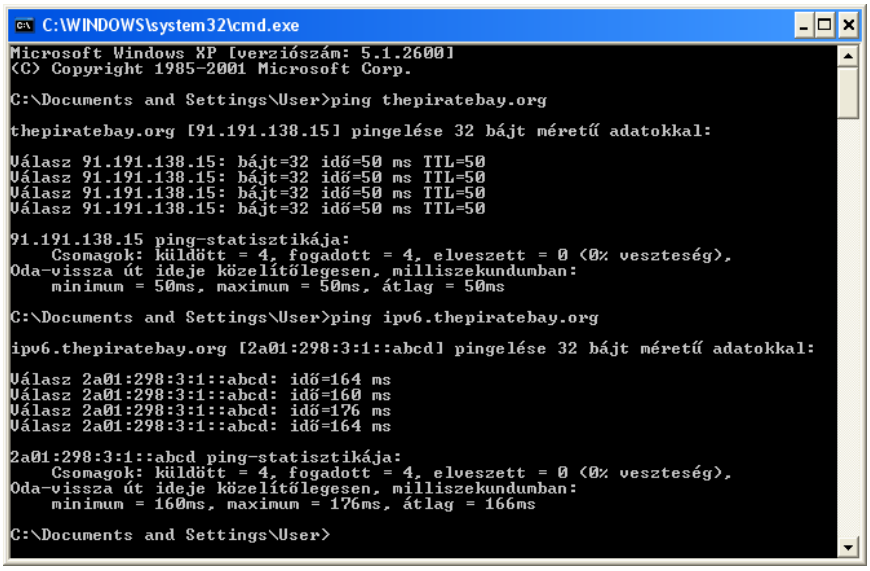
Azt IPv6-os címekre vonatkozó keresésekre létrehoztak egy új domaint, melynek célja az IPv6-os címek és hoszt nevek összerendelése. Ennek a domainnak a gyökere az IP6.ARPA. Egy IPv6-os cím „.IP6.ARPA” végződésű, pontokkal elválasztott 4 bites szavak sorozataként jelenik meg az IP6.ARPA domainban (pl.: ABCD:EF01:2345:6789:ABCD:EF01:2345:6789 esetén 9.8.7.6.5.4.3.2.1.0.F.E.D.C.B.A.9.8.7.6.5.4.3.2.1.0.F.E.D.C.B.A.IP6.ARPA). A 4 bites

szavak fordított sorrendben helyezkednek el, és minden egyes szó egy-egy hexadecimális számként van kódolva.

Mint már említettem, a kereséseket is megváltoztatták. Amennyiben egy keresés A rekordra vonatkozik,akkor az AAAA rekordokra is végre kell hajtani a keresést. Mivel a DNS kéréseknél is fontos a kapott adat hitelessége és megbízhatósága, ezért mindaddig nem biztonságosnak kell tekintenünk egy kérésből származó információt, amíg valamilyen DNS Security módszert nem alkalmaznak.

# 7 IPv6 a gyakorlatban

Mivel még napjainkban sincs Magyarországon olyan ISP, aki natív IPv6-os szolgáltatást üzemeltetne, ezért mi, felhasználók kénytelenek vagyunk különböző tunnelezési megoldásokat igénybe venni annak érdekében, hogy IPv6-on alapuló szolgáltatásokhoz jussunk. Ezek a szolgáltatások jelenleg a Web-böngészésben ki is merülnek ugyan, de például a világ legnagyobb bittorrent oldala, a thepiratebay.org már elindította az IPv6-os trackerét. A váltás szükségességének elismerése mellett sajnos azt is ki kell azonban jelenteni, hogy egy átlagos felhasználónak (sőt, akár egy nem átlagosnak is) semmi érdeke nem fűződik ahhoz, hogy IPv6-ot használjon, ugyanis a különböző tunnelezési eljárások általában nagy késleltetéssel járnak. Tekintsük például az előbb említett ThePirateBay.org címet, és pingeljük is meg azt:




A különbség azt hiszem látható, ugyanakkor nem jellemző, hogy ekkora legyen az eltérés. Ezek az eljárások ugyanis egytől-egyig úgy működnek, hogy becsomagolják az eredetileg IPv6-os csomagunkat egy IPv4-es csomagba, azt valamilyen úton eljuttatják a tunnelünk másik végére, ahol az kicsomagolódik, és immár IPv6-os csomagként folytatja útját a célig. Így a tunnelünk végén lévő szerver szűk keresztmetszetet adhat számunkra, ugyanis minden, az IPv6-os hálózatba tartó csomagunk keresztül megy rajta, és természetesen nem egyedül használjuk a szervert. Mivel a tunnel két végpontja között az IPv4-es hálózatot használjuk, ezért egy ilyen címre is szükség van, valamint ennek a hálózatnak is van egy késleltetése, ami

csak növeli az adatok átviteléhez szükséges időt. A tunnelek szemszögéből a felhasználók 3 különböző IPv4-es címmel rendelkezhetnek:

- Statikusan kiosztott
- Dinamikusan kiosztott
- Privát IP cím (NAT)

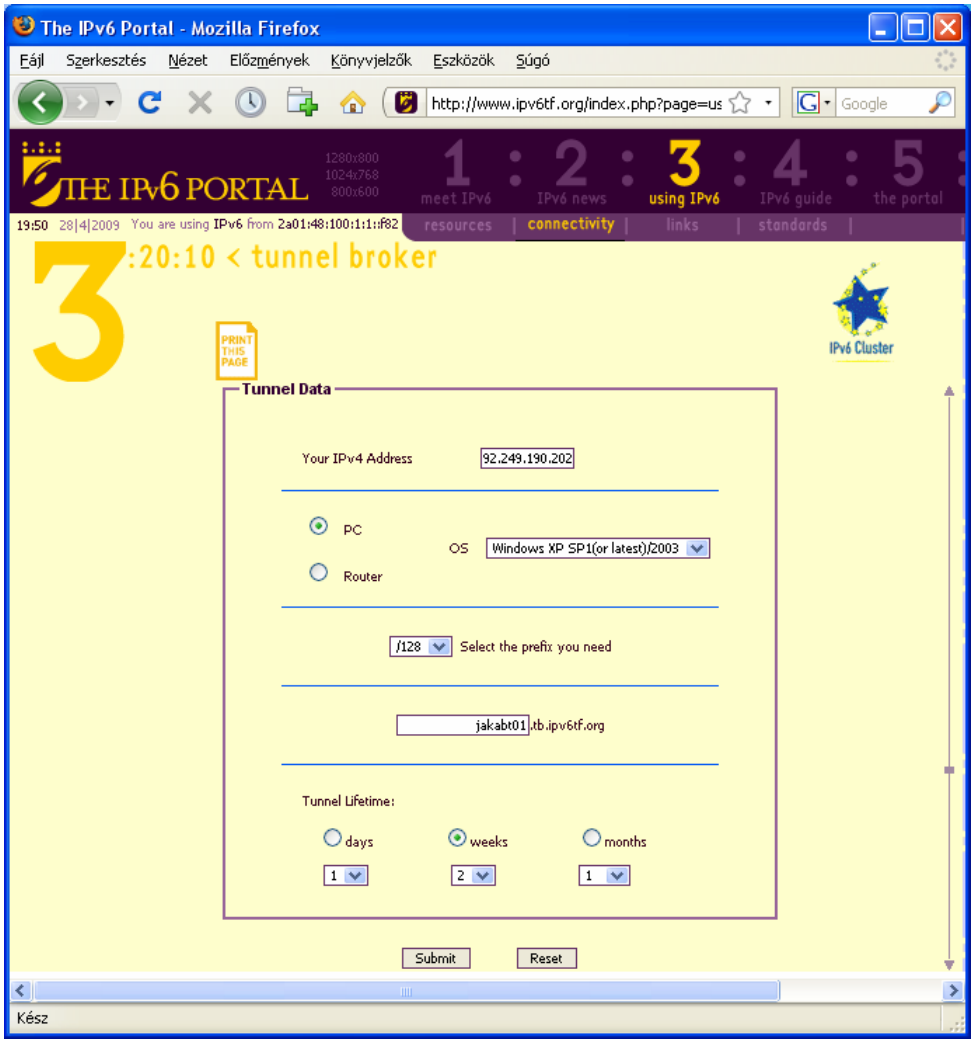
Azért fontos megkülönböztetni ezeket az eseteket, mert nem mindegy, hogy a szervernek milyen információkkal rendelkezik a tunnel másik végével kapcsolatban. Nézzük most meg tehát, hogy az egyes esetekben hogyan is épül fel, és működik az IPv6-os adatforgalom szállítására szolgáló alagút. Az IPv6-os támogatást Microsoft Windows XP operációs rendszert futtató számítógépen a netsh interface ipv6 install parancs kiadásával lehet telepíteni. A következőkben feltételezem, hogy ez már megtörtént.

## *7.1 Statikus IPv4-el működő tunnel (6in4)*

Ekkor egy kézzel definiált tunnelt hozunk létre, melyet a 4213-as RFC definiál. A küldő csomópont ekkor azt csinálja, hogy egy IPv4-es csomag adat részében juttatja el az IPv6-os csomagunkat a tunnel másik végén lévő csomóponthoz. Ekkor az IPv4-es fejléc mezőit a következő módon kell inicializálni. A típus 4-es értéket kap, az IP fejléc hossza 5 lesz, a ToS mezőt általában 0-ra állítják, az IPv4-es csomagban lévő adat pedig az IPv6-os csomag teljes hossza +az IPv4-es fejléc mérete. A darabolhatóságra vonatkozó opciókat természetesen szabadon állíthatja a feladó, attól függően, hogy mekkora az átvivendő adat, és az élettartam mezőt (TTL) is a megszokott módon kell inicializálni. A protokoll mező 41-es értéket kap, utalva ezzel arra, hogy a felsőbb réteg az IPv6. A forrás és cél címet pedig úgy kell beállítani, hogy azok a tunnel két végpontját azonosítsák. A tunnel végén a feldolgozásnál észreveszik, hogy a csomag egy IPv6-os datagramot hordoz, amit ezután tovább küldenek az IPv6-os fejlécnek megfelelően. A tunnel két végpontja között akármilyen bonyolultságú hálózat lehet, az IPv6- szempontjából az csak ez szimpla pont-pont kapcsolatnak látszik, hiszen a tunnel elrejti az IPv4-es hálózatra vonatkozó információkat az IPv6 elől. Fontos még megjegyezni, hogy amennyiben a kicsomagolásnál azt tapasztalják, hogy egy érvénytelen IPv6-os címet tartalmaz a csomag, akkor azt némán el kell dobni. Bővebb információt a fent említett RFC dokumentumban talál az olvasó erről a szolgáltatásról.

Nézzük most akkor működés közben:

Először is szükségünk van egy tunnel brókerre, aki biztosítja az IPv6-os alagút másik végét. Én az ipv6tf.org-ot választottam. Rövid regisztráció után nincs más dolgunk, mint igényelni egy tunnelt:




Miután ezzel készen vagyunk, már csak a saját gépünkön kell konfigurálni a tunnel végpontját, és már működik is. Az ehhez szükséges parancsok pedig a következők:

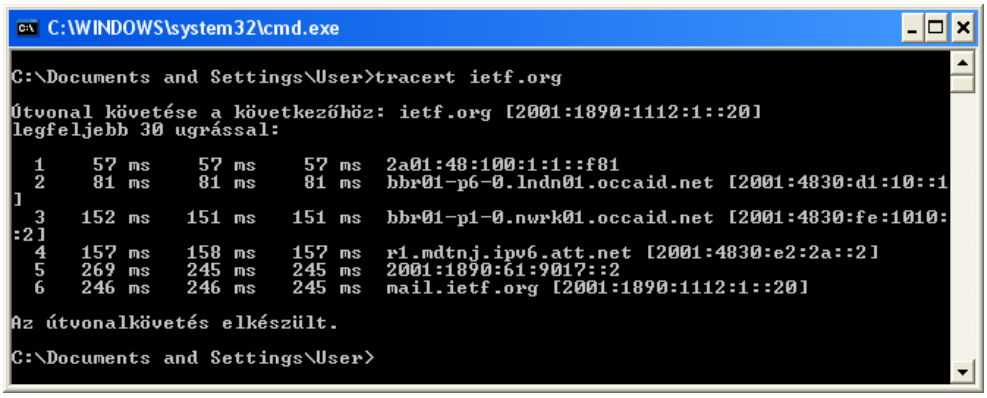
netsh interface ipv6 add v6v4tunnel Consulintel 92.249.190.202 213.172.34.125

netsh interface ipv6 set interface Consulintel mtu=1480

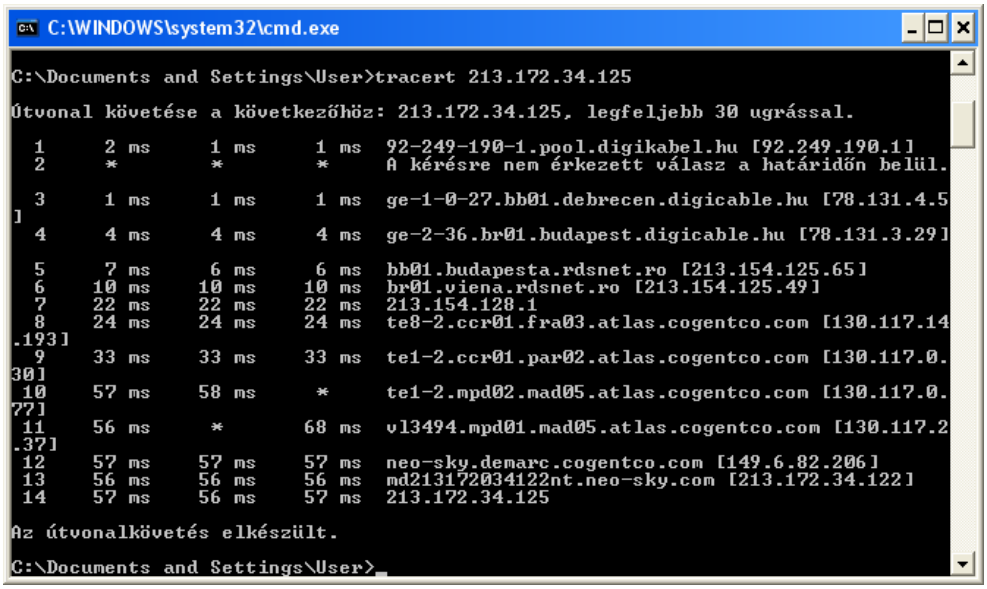
netsh interface ipv6 add address Consulintel 2A01:0048:0100:0001:0001::F82

netsh interface ipv6 add route 0::/0 Consulintel 2A01:0048:0100:0001:0001::F81 store=persistent

Ezek után a tracert parancsot kiadva a következőket tapasztaljuk:



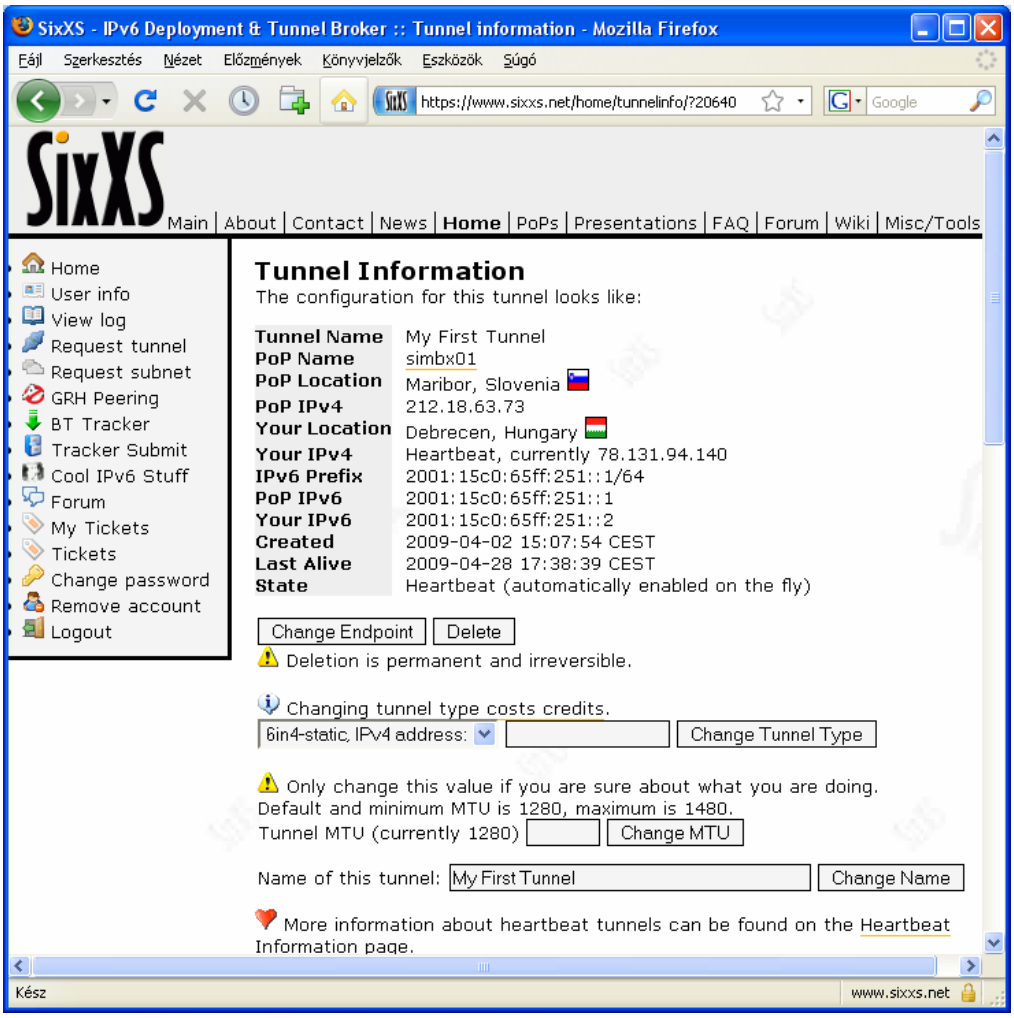

Lássuk ezek után, hogy mit rejt a tunnel:



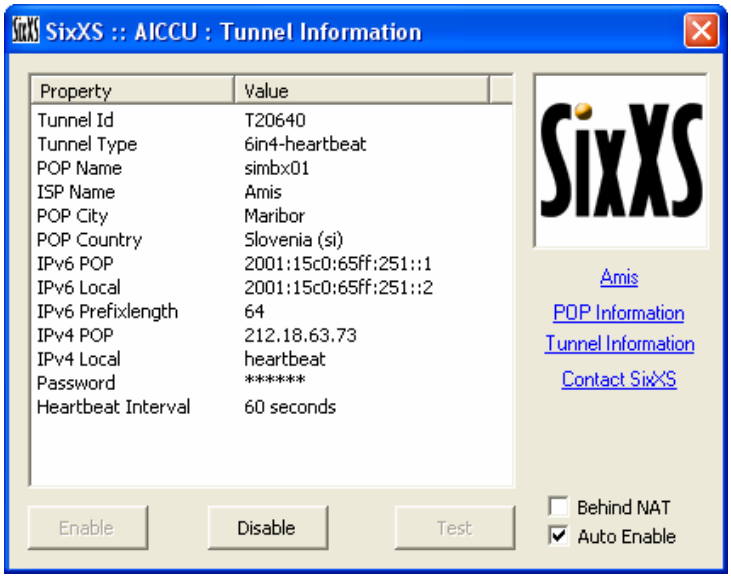

## *7.2 Dinamikus IPv4-el működő tunnel (Heartbeat)*

Ennek a módszernek az a különlegessége az előzőhöz képest, hogy a felhasználó dinamikus IPv4-es címet kap az ISP-ről, aminek következtében a tunnel egyik vége mozogni fog annak megfelelően, hogy mi az aktuális IPv4-es címe a felhasználónak. Ezt a mozgást egy UDP csomag küldésével követik, és konfigurálják újra a tunnelt úgy, hogy például böngészés közben ebből semmit nem veszünk észre. Alapját egy úgynevezett Heartbeat protokoll alkotja, ami meghatározott időközönként küld a szervernek egy-egy UDP csomagot, benne a végpontra vonatkozó információkkal. Az UDP csomagban természetesen hitelesítési információk is szerepelnek, megakadályozva ezzel az illetéktelen felhasználást. Az UDP csomag szerkezetére és a protokoll működésére vonatkozó további információt a [9]-ben találhat az Olvasó. Végfelhasználóként talán ezt a legegyszerűbb használni, ugyanis csak

annyi dolgunk van, hogy regisztrálunk a www.sixxs.net honlapon, igénylünk egy tunnelt, letöltjük a segédprogramot, ami képes szolgáltatásként beépülni a windowsba, és annak indulásakor elindulni. Miután regisztráltunk, és megkaptuk az igényelt tunnelt, a következő látvány fogad minket:




A segédprogram neve aiccu-2006-07-23-windows-gui.exe, letöltése után egy bejelentkező képernyőt követően, a tunnelünket kiválasztva elérhetjük a konfigurációs paramétereket:




Mint látható, a Heartbeat protokoll és az AICCU segédprogram segítségével képesek lennénk akár NAT mögül is élvezni az IPv6 szolgáltatásait, ezért eltekintenék a harmadik lehetőség részletezésétől, azonban a kíváncsi Olvasó a [10] dokumentumban részletesen olvashat erről a módszerről.

## *7.3 Socket programozás*

Dolgozatom során egy fontos dologról nem beszéltem még. Mégpedig arról, hogy milyen változásokat hoz az IPv6 bevezetése a socket programozásban. Ebben a fejezetben ezt a hiányosságot szeretném kiküszöbölni. Mivel az IPv4-es és IPv6-os cím mérete eltér, ezért az adatstruktúrákat át kellett dolgozni annak érdekében, hogy kezelni tudják a 128 bites címeket.

A fontosabb adatstruktúrák deklarációja a következő:

```
typedef struct in6_addr {
  union {    u_char Byte[16];    u_short Word[8];  } u;
} IN6_ADDR, *PIN6_ADDR, FAR *LPIN6_ADDR;

struct sockaddr_in6 {
        short   sin6_family;
        u_short sin6_port;
        u_long  sin6_flowinfo;
        struct  in6_addr sin6_addr;
        u_long  sin6_scope_id;
};

typedef struct sockaddr_in6 SOCKADDR_IN6;
typedef struct sockaddr_in6 *PSOCKADDR_IN6;
typedef struct sockaddr_in6 FAR *LPSOCKADDR_IN6;

typedef struct in_pktinfo {
  IN6_ADDR ipi6_addr;
```

```
  UINT     ipi6_ifindex;
} in_pktinfo;
typedef struct ipv6_mreq {
  struct in6_addr ipv6mr_multiaddr;
  unsigned int    ipv6mr_interface;
} IPV6_MREQ, *PIPV6_MREQ;
```

Nézzünk most egy rövidke kódot, mely szemlélteti, hogyan épül fel egy IPv6-os socket:

```
//Változók deklarálása
int kapu_figyelo, sockoptval=1;
struct sockaddr_in6 szerver;
memset((void *)&szerver, 0, sizeof(szerver));
szerver.sin_family= AF_INET6;
inet_aton("::1", &(szerver.sin6_addr));
szerver.sin6_port= htons(5150);

//hallgatózó socket létrehozása
if((kapu_figyelo = socket(AF_INET6, SOCK_STREAM, IPPROTO_IPV6))== -1 ){
        perror("socket");
        exit(EXIT_FAILURE);
}
setsockopt(kapu_figyelo, SOL_SOCKET, SO_REUSEADDR,(void *)&sockoptval,
sizeof(sockoptval));

//a hallgatózó socket és a helyi cím összerendelése
if(bind(kapu_figyelo, (struct sockaddr_in6 *)&szerver,sizeof(szerver)) == -
1){
        perror("bind");
        exit(EXIT_FAILURE);
}
```

Ezzel létrehoztunk egy kommunikációra kész IPv6-os socketet, amely az 5150-es porton hallgatózik. Ezután már csak a `connect()` függvénnyel csatlakozni kell hozzá, amit egy `accept()` függvénnyel elfogadunk, és máris mehet az üzenetküldés. Mivel azonban a kommunikáció során használatos függvényeket nem érintette a változás, ezért ezeket nem részletezném.

# Összefoglaló

A technológia fejlődésével a számítógépek és az internet egyre nagyobb teret töltenek be. Egyre több ember használja nap-mint nap a világhálót, más és más célból. Ebből kifolyólag elkerülhetetlennek látszik, hogy előbb-utóbb elérjük a több tíz éves IPv4-es címzési rendszer teljesítőképességének a határát. Megoldás van, már az is elmúlt tíz éves, de a váltást még mindig nem sikerült a szakmának kierőszakolnia. Dolgozatomban bemutattam az új generációs IP címet, az IPv6-ot, részletesen tárgyaltam az IPv6-os fejlécet, valamint a további kiterjesztett fejléceket is. Ahol szükséges volt, kitértem a biztonsági kockázatokra, igyekeztem példákkal is szemléltetni a működést. Mivel az internet működéséhez nem csak egy címre van szükség, tárgyaltam azokat a kiegészítő, ám nem elhanyagolható megoldásokat, mint például a DNS, vagy az ICMPv6, ami nélkül gyakorlatilag lehetetlen lenne a hibadetektálás. Szót ejtettem továbbá olyan fontos mechanizmusokról, mint a Neighbor Discovery, ami lehetővé teszi az automatikus konfigurációt, és olyan fontos szerepeket lát el, mint az előző verzióban az ARP. A téma nagyságából adódóan igyekeztem egy minimális, ugyanakkor kellően alapos leírást adni az IPSec-ről, hiszen manapság már nem képzelhető el hálózati kommunikáció valamilyen authentikációs folyamat nélkül.

A téma egységességéhez hozzá tartoznak a különböző tunnelezési megoldások is, hiszen Magyarországon kizárólag ilyen módon juthatunk IPv6-os címhez, és véleményem szerint ez a közeljövőben nem is fog változni. Végül, de nem utolsó sorban a hálózati kommunikáció programozásában bekövetkező változásokat mutattam be, hiszen mit sem ér az egész, ha nem íródnak olyan programok, amik használják a bemutatott eszközrendszert.

A dolgozat megírásakor az a cél vezérelt, hogy egy átfogó, mindenki számára érthető munkát készítsek, ami rávilágít a váltás szükségességére és előnyeire, betekintést enged a színfalak mögé, és hogy egy laikus felhasználónak is útmutatóul szolgálhasson az IPv6 használatához.

*Köszönetnyilvánítás*

*Ezúton szeretném megköszönni témavezetőmnek, Dr. Almási Bélának, hogy szakirodalmakkal és értékes tanácsaival elősegítette e mű elkészülését!*

# Irodalomjegyzék


[1] Andrew S Tanenbaum: Számítógép Hálózatok, Budapest: Panem, [2008] ISBN 963 545 384 1

[2] http://www.iana.org/assignments/ipv4-tos-byte

[3] http://www.secdev.org/conf/IPv6_RH_security-csw07.pdf

[4] http://wiki.sch.bme.hu/pub/Villanyszak/Eszolgaltatas/Kommunikaciobiztonsaga.pdf

[5] http://en.wikipedia.org/wiki/6in4

[6] http://en.wikipedia.org/wiki/IPsec

[7] http://feczo.nmi.rulez.org/vpn/szakdolgozat/

[8] http://ipv6.niif.hu/tipster6/papers/overview/3.fejezet.html

[9] http://tools.ietf.org/html/draft-massar-v6ops-heartbeat-01

[10] http://unfix.org/~jeroen/archive/drafts/draft-massar-v6ops-ayiya-02.txt

[11] R. Hinden and S. Deering, „IP Version 6 Addressing Architecture”, RFC 4291, February 2006.

[12] A. Conta, S. Deering and M. Gupta, Ed., „Internet Control Message Protocol (ICMPv6) for the Internet Protocol Version 6 (IPv6) Specification”, RFC 4443, March 2006.

[13] J. Loughney, Ed., „IPv6 Node Requirements”, RFC 4294, April 2006.

[14] R. Hinden, S. Deering and E. Nordmark, „IPv6 Global Unicast Address Format”, RFC 3587, August 2003.

[15] J. Abley, P. Savola and G. Neville-Neil, „Deprecation of Type 0 Routing Headers in IPv6”, RFC 5095, December 2007.

[16] S. Thomson, C. Huitema, V. Ksinant and M. Souissi, „DNS Extensions to Support IP Version 6”, RFC 3596, October 2003.

[17] S. Kent and K. Seo, „Security Architecture for the Internet Protocol”, RFC 4301, December 2005.

[18] S. Kent, „IP Authentication Header”, RFC 4302, December 2005.

[19] S. Kent, „IP Encapsulating Security Payload (ESP)”, RFC 4303, December 2005.

[20] D. Harkins and D. Carrel, „The Internet Key Exchange (IKE)”, RFC 2409, November 1998.

[21] P. Savola and B. Haberman, „Embedding the Rendezvous Point (RP) Address in an IPv6 Multicast Address”, RFC 3956, November 2004.

[22] T. Narten, E. Nordmark, W. Simpson and H. Soliman, „Neighbor Discovery for IP version 6 (IPv6)”, RFC 4861, September 2007.

[23] E. Nordmark and R. Gilligan, „Basic Transition Mechanisms for IPv6 Hosts and Routers”, RFC 4213, October 2005.

[24] B. Haberman and D. Thale, „Unicast-Prefix-based IPv6 Multicast Addresses”, RFC 3306, August 2002.

[25] http://beej.us/guide/bgnet/output/html/singlepage/bgnet.html

[26] http://msdn.microsoft.com/en-us/library/ms741416(VS.85).aspx